공학박사 학위논문

Design and Application of Deep-learning Artificial Neural Network for Human Motion and Posture Analysis

인체 동작 및 자세 분석을 위한 심화 학습 인공신경망 설계 및 적용

2020 년 7 월

서울대학교 대학원

산업·조선공학부 인간공학 전공

진 병 기

Abstract

Design and Application of Deep-learning Artificial Neural Network for Human Motion and Posture Analysis

Byungki Jin

Department of Industrial Engineering

The Graduate School

Seoul National University

Ergonomic research is conducted through observation, measurement, and analysis. Ergonomic research has also been developed due to the development of technologies related to observation, measurement, and analysis. Deep-learning technology is a core technology for artificial intelligence development. Various attempts have been made to complement and replace human capabilities like observation, measurement, and analysis, using deep-learning technologies. This deep-learning technology can be applied to various stages of the ergonomic research process. Therefore, in this research, various attempts were made to prepare methods for applying deep-learning to ergonomic research.

This thesis attempted to analysis via deep-learning to various kinds of data, such as numerical data, image data, and video data. Besides, to identify the characteristics of data that can be applied to deep-learning, different data collecting methods were applied. The data types were data collected for deep-learning, data collected without considering deep-learning, and data collected and released by the government.

The first research is to detect sitting posture from body pressure distribution data. Back health is closely related to the user's sitting posture, so it is crucial to have a good sitting posture when young. In a controlled environment, body pressure distribution image data for seven postures were collected from children. The deep-learning method used for posture classification is a convolutional neural network (CNN). The classification performance of logistic regression and CNN is compared. As a result, CNN showed a 20% improvement over logistic regression in the overall classification performance.

The second research is to derive work risk assessments using assembly process videos. The data used in the study were those used in the work risk assessment. The performance was evaluated by applying LSTM, one of the deep-learning methods, to the work risk assessment methods OWAS, RULA, and REBA. As a result, when performing OWAS with deep-learning, it showed better performance than RULA and REBA.

The third research estimates the stature from hand dimensions. The data used in this research were investigated and released by the government. In the previous study, the stature was estimated from hand dimensions using linear regression. Linear regression, RNN, and the recursive generalized linear model (RGLM) were applied to compare the performance of stature estimation. As a result, deep learning techniques RNN and RGLM performed better than linear regression.

Through three research, it was confirmed that the deep-learning method could replace the existing research method. Although the absolute performance was not excellent, it showed relatively good performance than the existing method. The deep-learning method was different depending on the data format and condition. The performance difference also occurred according to the kind of

deep-learning method. If the various cases were not learned, no results were obtained for the missing parts. Therefore, data selection and pre-processing must be preceded while applying deep-learning. In ergonomic research, deep-learning will make it easy to reflect the results of ergonomic research into reality. Deep-learning will not replace the researcher but will broaden the research subject's scope and make the research results widely available.

Keywords: Human Factors, Deep Learning , Ergonomics, Posture Recognition, Motion Analysis, Regression Model

Student Number: 2010-21121

Contents

Abstract **i**

Contents **viii**

List of Tables **xiii**

List of Figures **xvii**

Chapter 1 Introduction **1**

1.1 Research Background 1

1.2 Purpose of This Study 4

1.3 Organization of the Thesis 5

Chapter 2 Literature Reviews **9**

2.1 Sitting Posture 9

2.2 Working Posture Measurement 15

2.3 Anthropometric Dimension Estimation 20

2.4 Deep-learning Application 22

Chapter 3 An Ergonomic Analysis of Seated Posture using a Deep-learning Method 25

3.1 Overview . . . 25
3.2 Data Characteristics . . . 26
3.2.1 Body Pressure Distribution on Seat Cushion . . . 26
3.2.2 Data Collection . . . 27
3.2.3 Data Pre-processing . . . 29
3.3 Data Analysis . . . 32
3.3.1 Convolutional Neural Network . . . 32
3.3.2 Performance Comparison Method . . . 34
3.4 Results . . . 36
3.4.1 Logistic Regression . . . 36
3.4.2 Convolutional Neural Networks . . . 39
3.4.3 Comparison of Logistic Regression Results and Convolutional Neural Networks Results . . . 42
3.5 Discussion . . . 44

Chapter 4 Applying Deep-learning Methods to Human Motion Analysis of Automobile Assembly Tasks 47

4.1 Overview . . . 47
4.2 Data Characteristics . . . 48
4.2.1 Work-related Musculoskeletal Disorders(WMSDs) in Factory Workers . . . 48
4.2.2 Data Collection . . . 49
4.2.3 Data Pre-processing . . . 50
4.3 Data Analysis . . . 52
4.4 Results . . . 52
4.4.1 OWAS Prediction Model . . . 52

4.4.2 RULA Prediction Model 53
4.4.3 REBA Prediction Model 54
4.5 Discussion 55

Chapter 5 Estimation of Hand Anthropometric Dimensions Using a Deep-learning Method 59

5.1 Overview 59
5.2 Data Characteristics 60
5.2.1 Size Korea; A National Anthropometric Survey of Korea . 60
5.2.2 Hand Anthropometric Measurement Data 61
5.2.3 Data Selection and Hand Dimension 62
5.2.4 Training Data and Test Data 64
5.3 Data Analysis 65
5.4 Result 66
5.4.1 Comparison of Relative Absolute Error(RAE) 68
5.4.2 Comparison of Relative Squared Error(RSE) 70
5.4.3 Comparison of Mean Absolute Percentage Error(MAPE) . 72
5.4.4 Comparison of Mean Absolute Scaled Error(MASE) . . . 74
5.4.5 Comparison of Root Mean Square Error(RMSE) 76
5.4.6 Comparison of Mean Absolute Error(MAE) 78
5.4.7 Comparison of Mean Squared Error(MSE) 80
5.4.8 Clustering the Results Along with the Performance 82
5.5 Discussion 83

Chapter 6 Discussion and Conclusions 87

6.1 Summary of findings 87
6.2 Contributions of this study 89
6.3 Limitations and further studies 92

Bibliography **95**

Appendix A Confusion Matrix from Chapter III **104**

Appendix B Python Code for Chapter III **125**

Appendix C Python Code for Chapter IV **129**

Appendix D Python Code for Chapter V **141**

국문초록 **145**

List of Tables

Table 2.1 Previous studies that predicts sitting postures 14

Table 2.2 Action categories(AC) in OWAS from Karhu et al. (1977) 16

Table 3.1 Confusion matrix for the relationship between actual values and classification results 35

Table 3.2 F1 score and accuracy of cross-validation results on logistics regression 37

Table 3.3 It is a confusion matrix that sums up the results of cross-validation using logistic regression. 38

Table 3.4 F1 score and accuracy of cross-validation results of convolutional neural networks 40

Table 3.5 It is a confusion matrix that sums up the results of cross-validation using Convolutional Neural Networks. . . 41

Table 3.6 Descriptive statistics on accuracy values for each estimation method. 43

Table 3.7 Paired samples t-tests result of accuracy value 43

Table 4.1 Images and evaluation results collected during the years of musculoskeletal hazard investigations on automobile factory workers 49
Table 4.2 Confusion Matrix of OWAS Action Category(AC) Prediction 53
Table 4.3 Confusion Matrix of RULA Score Prediction 53
Table 4.4 Confusion Matrix of REBA Score Prediction 54
Table 4.5 OWAS, RULA, REBA score distribution of data used for training and test 56
Table 4.6 Ratio of data set and classification result of the top two items of OWAS, RULA, and REBA 57

Table 5.1 Hand dimensions measured in Size Korea project 61
Table 5.2 List of parts for human body measurement 62
Table 5.3 List of 29 hand regions selected to estimate stature in Jee and Yun (2015) 63
Table 5.4 Regression performance evaluation metrics 65
Table 5.5 RAE, RSE, MAPE, MASE, RMSE, MAE, MSE value of CNN, RNN, RGLM 67
Table 5.6 Descriptive statistics on RAE values for each estimation method. 68
Table 5.7 Paired samples t-tests result of RAE value 68
Table 5.8 Descriptive statistics on RSE values for each estimation method. 70
Table 5.9 Paired samples t-tests result of RSE value 70
Table 5.10 Descriptive statistics on MAPE values for each estimation method. 72
Table 5.11 Paired samples t-tests result of MAPE value 72

Table 5.12 Descriptive statistics on MASE values for each estimation method. 74
Table 5.13 Paired samples t-tests result of MASE value 74
Table 5.14 Descriptive statistics on RMSE values for each estimation method. 76
Table 5.15 Paired samples t-tests result of RMSE value 76
Table 5.16 Descriptive statistics on MAE values for each estimation method. 78
Table 5.17 Paired samples t-tests result of MAE value 78
Table 5.18 Descriptive statistics on MSE values for each estimation method. 80
Table 5.19 Paired samples t-tests result of MSE value 80
Table 5.20 Clustering result of performance comparison by methods . 82

Table A.1 Logistic Regression, Cross-validation No. 1 105
Table A.2 CNN, Cross-validation No. 1 105
Table A.3 Logistic Regression, Cross-validation No. 2 105
Table A.4 CNN, Cross-validation No. 2 106
Table A.5 Logistic Regression, Cross-validation No. 3 106
Table A.6 CNN, Cross-validation No. 3 106
Table A.7 Logistic Regression, Cross-validation No. 4 107
Table A.8 CNN, Cross-validation No. 4 107
Table A.9 Logistic Regression, Cross-validation No. 5 107
Table A.10 CNN, Cross-validation No. 5 108
Table A.11 Logistic Regression, Cross-validation No. 6 108
Table A.12 CNN, Cross-validation No. 6 108
Table A.13 Logistic Regression, Cross-validation No. 7 109
Table A.14 CNN, Cross-validation No. 7 109

Table A.15 Logistic Regression, Cross-validation No. 8 109
Table A.16 CNN, Cross-validation No. 8 110
Table A.17 Logistic Regression, Cross-validation No. 9 110
Table A.18 CNN, Cross-validation No. 9 110
Table A.19 Logistic Regression, Cross-validation No. 10 111
Table A.20 CNN, Cross-validation No. 10 111
Table A.21 Logistic Regression, Cross-validation No. 11 111
Table A.22 CNN, Cross-validation No. 11 112
Table A.23 Logistic Regression, Cross-validation No. 12 112
Table A.24 CNN, Cross-validation No. 12 112
Table A.25 Logistic Regression, Cross-validation No. 13 113
Table A.26 CNN, Cross-validation No. 13 113
Table A.27 Logistic Regression, Cross-validation No. 14 113
Table A.28 CNN, Cross-validation No. 14 114
Table A.29 Logistic Regression, Cross-validation No. 15 114
Table A.30 CNN, Cross-validation No. 15 114
Table A.31 Logistic Regression, Cross-validation No. 16 115
Table A.32 CNN, Cross-validation No. 16 115
Table A.33 Logistic Regression, Cross-validation No. 17 115
Table A.34 CNN, Cross-validation No. 17 116
Table A.35 Logistic Regression, Cross-validation No. 18 116
Table A.36 CNN, Cross-validation No. 18 116
Table A.37 Logistic Regression, Cross-validation No. 19 117
Table A.38 CNN, Cross-validation No. 19 117
Table A.39 Logistic Regression, Cross-validation No. 20 117
Table A.40 CNN, Cross-validation No. 20 118
Table A.41 Logistic Regression, Cross-validation No. 21 118

Table A.42 CNN, Cross-validation No. 21 118
Table A.43 Logistic Regression, Cross-validation No. 22 119
Table A.44 CNN, Cross-validation No. 22 119
Table A.45 Logistic Regression, Cross-validation No. 23 119
Table A.46 CNN, Cross-validation No. 23 120
Table A.47 Logistic Regression, Cross-validation No. 24 120
Table A.48 CNN, Cross-validation No. 24 120
Table A.49 Logistic Regression, Cross-validation No. 25 121
Table A.50 CNN, Cross-validation No. 25 121
Table A.51 Logistic Regression, Cross-validation No. 26 121
Table A.52 CNN, Cross-validation No. 26 122
Table A.53 Logistic Regression, Cross-validation No. 27 122
Table A.54 CNN, Cross-validation No. 27 122
Table A.55 Logistic Regression, Cross-validation No. 28 123
Table A.56 CNN, Cross-validation No. 28 123
Table A.57 Logistic Regression, Cross-validation No. 29 123
Table A.58 CNN, Cross-validation No. 29 124

List of Figures

Figure 1.1 Organization of the thesis 7

Figure 2.1 The appearance of lumbar spine according to posture . . 10

Figure 2.2 Quartiles of breaks in sedentary time with metabolic risk variables: waist circumference, BMI. (Healy et al., 2008) 10

Figure 2.3 A procedure to recognize the actual person's posture by comparing the figure of the person who was photographed with the camera and the 3D model (Boulay et al., 2005) 11

Figure 2.4 Using the infrared marker attached at the position of (a), the position of the sensor suitable for posture monitoring was derived and the garment was made as shown in (b). (Dunne et al., 2008) 12

Figure 2.5 Classification accuracy for "familiar" and new subjects. Solid, dashed and dotted lines correspond to accuracy associated with the smallest ε value, first two smallest ε values, and the first three smallest ε values, respectively (Tan et al., 2001). 13

Figure 2.6 Standard working postures in the OWAS method (Luopajarvi, 1990) 16

Figure 2.7 RULA employee assessment worksheet (Hedge, 2000b) . 18

Figure 2.8 REBA employee assessment worksheet (Hedge, 2000a) . 19

Figure 2.9 Layer structure of LeNet-5(LeCun et al., 1998) 23

Figure 2.10 Scatter plot for one-day flow forecasting (Le et al., 2019). 23

Figure 3.1 The component of data collection system for the experiment. (a) Chair cushion with pressure sensor, (b) android smartphone environment for receiving pressure data, (c) Duoback RA-070SDSF 27

Figure 3.2 Seven sitting postures used in data collection. (a) Sitting straight, (b) Lean forward, (c) Lean left, (d) Lean right, (e) Lean backward, (f) Sitting at the front of the chair, (g) Sitting crossed-legged on the chair 28

Figure 3.3 Example of sensor data preprocecing. (The size of the rectangle in (d) reflects the actual size of data reduction.) 31

Figure 3.4 Layer structure of LeNet-5(LeCun et al., 1998) 33

Figure 3.5 Layer structure of modified LeNet-5 model in this study 34

Figure 3.6 Accuracy value according to cross-validation applying Logistic regression 36

Figure 3.7 Average of precision, recall, and F1 score for each sitting posture. (a) Sitting straight, (b) Lean forward, (c) Lean left, (d) Lean right, (e) Lean backward, (f) Sitting at the front of the chair, (g) Sitting crossed-legged on the chair 38

Figure 3.8 Accuracy value according to cross-validation applying Convolutional Neural Networks 39

Figure 3.9 Average of precision, recall, and F1 score for each sitting posture. (a) Sitting straight, (b) Lean forward, (c) Lean left, (d) Lean right, (e) Lean backward, (f) Sitting at the front of the chair, (g) Sitting crossed-legged on the chair 42

Figure 3.10 Cross-validation results comparison between logistic regression and Convolutional neural networks 42

Figure 3.11 95% Confidence Interval and Median Distribution Chart for Accuracy Values 44

Figure 4.1 The process of extracting human body motion from working video: (a) Working video, (b) OpenPose applied video, (c) Background removed video, (d) Only human motion saved as JSON. 51

Figure 5.1 Various hand landmarks for measurement (Jee and Yun, 2015) 64

Figure 5.2 95% Confidence Interval and Median Distribution Chart for RAE Values 69

Figure 5.3 95% Confidence Interval and Median Distribution Chart for RSE Values 71

Figure 5.4 95% Confidence Interval and Median Distribution Chart for MAPE Values 73
Figure 5.5 95% Confidence Interval and Median Distribution Chart for MASE Values 75
Figure 5.6 95% Confidence Interval and Median Distribution Chart for RMSE Values 77
Figure 5.7 95% Confidence Interval and Median Distribution Chart for MAE Values 79
Figure 5.8 95% Confidence Interval and Median Distribution Chart for MSE Values 81

Chapter 1

Introduction

1.1 Research Background

Historical records of ergonomics go back to ancient Greece(Marmaras et al., 1999). Greek civilization in the fifth century BC used ergonomic principles in the design of their tools, jobs, and workplaces. Then in the late nineteenth century, Frederic Winslow Taylor began the study of ergonomics in the modern sense. He used a stopwatch in the Time Study to measure the minimum time of operation and then calculated the appropriate daily production (Taylor, 1903). At the time, the best observation was to observe with eyes, the best equipment to measure time was a stopwatch, and the best equipment to calculate daily yield was his hands and head. Since then, Frank & Lillian Gilbreth has used a camera in Motion Study to improve human behavior. As such, the emergence of new observation equipment, measuring equipment, and calculating equipment has led to the development of ergonomics.

Observing humans has been in sync with the development of equipment

related to photography. The researchers first relied on the eyes. Then the technology developed with black and white photographs, black and white videos, color photographs, color videos. Due to the development of digital technology, this technology has developed dramatically. As the limit of shooting time is gone, all human actions can be recorded. Besides, due to the development of computer technology, it has become possible to record human behavior in 3D data using motion capture technology. Also, the invention of the eye tracker and smart glasses made it possible to observe the first-person view rather than the third-person view. The development of various sensor technologies has broadened the scope of observation, including body temperature, heart rate, and number of steps, as well as our visual observation.

The development of measurement equipment has evolved into a direction in which the accuracy of measurement is increased, and objects of measurement are increasing. The boundary between measurement technology and observation technology has become blurred. Earlier researchers measured the length and time. However, now the object of measurement is significantly increased by electromyography(EMG), galvanic skin response(GSR), electroencephalography(EEG), electrocardiogram(ECG/EKG), respiration. This is a result of the combined development of various technologies such as electronic technology, sensor technology, computer technology, and miniaturization technology. As measurement accuracy increases, minute changes can be measured and even for areas that cannot be observed using conventional methods. Advances in computer and sensor technology have made it possible to measure static data as well as time-series data.

To process this massive amount of data and time-series data, the researchers used statistical tools such as SPSS and SAS, to process the data. Thus, many

methods, such as discriminant analysis and logistic regression, became popular. However, these methodologies also have limitations. Principal among these is their dependence on a fixed, underlying model or functional form. Discriminant analysis uses linear summation of independent variables to differentiate one category from another (Huberty and Lowman, 1997). Logistic regression also makes use of linear summations of independent variables, incorporated into a logistic function(Myers, 1990). Koza et al. (1999) observed that both techniques use regression merely to discover numerical coefficients for predetermined models. However, this parametric model may not be a suitable model when the result cannot be predicted or classified based on simple linear summation of independent variables.

Machine learning research offers the human factors professional a viable alternative for classification model development. Decision tree induction and genetic programming are two of these machine learning approaches. Unlike the traditional statistical methods, these methods do not rely on predetermined models using linear summations of independent variables. Decision tree induction has been used in the past to identify common patterns associated with specific automotive accident outcomes (Sohn and Shin, 2001). Genetic programming has been used to discover solutions to various ergonomics design problems involving control panels (Pham and Onder, 1992), and lifting tasks (Carnahan and Redfern, 1998a). Carnahan and Redfern (1998b) have used genetic programming to develop models that accurately classify lifting tasks as posing high or low risk of occupational back injury.

Machine learning techniques have evolved to create various methods such as K-nearest neighbors algorithm (KNN), principal component analysis (PCA), hidden Markov models (HMM), and support vector machines (SVM). Among

them, artificial neural network (ANN) based on human neurons developed into deep-learning by the development of optimization technology and computer technology. Deep-learning, which extracts features of data and optimizes hierarchically, has become a more accurate technology with the help of big data. AlphaGo has already won the go-go competition with humans using deep-learning technology. Deep-learning technology is a core technology of artificial intelligence, and its utilization is increasing compared to existing machine learning. Therefore, this study applies this deep-learning technique to the ergonomics research, reveals its limitations, and seeks to utilize it.

1.2 Purpose of This Study

The purpose of this study is to derive new methods of using ergonomic research data by applying deep-learning techniques to ergonomic research. Depp-learning is the core technology of artificial intelligence. It is designed to reflect not only linear but also nonlinear phenomena. In this study, the deep-learning technique is applied to numerical data, image data, and video data. The methodologies for the data format were derived, and the benefits and limitations were drawn.

First, the ergonomic application of data obtained from IoT (internet of things) products which will be used in real life is conducted. Ergonomic researchers used sensor data collected in real-time for various studies. Advances in sensor technology, communication technology, and computer technology make it easy to collect ergonomic data from various sensors attached to everyday products. In this study, the deep-learning technique was applied to estimate the sitting position in the body pressure distribution data of the chair seat.

Second, the deep-learning technique is applied to the analysis of physical tasks that are difficult to automate and analyzed whether automation is pos-

sible. Video is an excellent form of data for observing human behavior. Automating this video data analysis can save time and money. The deep-learning technique was used to determine the degree of musculoskeletal hazard from the operator's video. The method of applying the deep-learning technique to the video data was derived, and its practicality and limitations were examined.

Third, the deep-learning technique was applied to the analysis of body size information. Body size information is representative numerical data used in ergonomic research. In this study, the deep-learning technique was applied to the hand size data, which was handled by various methods such as linear regression. In order to measure and compare the performance of the deep-learning technique, various regression performance evaluation metrics were applied and compared with linear regression.

By applying deep-learning techniques to images, video, and numerical data used in ergonomics research, ergonomics research can take a step further. Deep-learning is not intended to replace ergonomic researchers with computers. Only ergonomic researchers can answer what data to collect and how to interpret it. To prepare for this change of time is the final purpose of this study.

1.3 Organization of the Thesis

The composition of This thesis can be summarized in Figure 1.1.

In Chapter 1, the research background is used to define the problem, and the purpose of this study is presented.

Chapter 2 deals with literature research related to this study. The researches related to sitting posture estimation, working posture measurement, and anthropometric dimension estimation were studied. Besides, a literature study

on the application of deep-learning, the primary tool of this study, was also conducted.

Chapter 3 uses deep-learning to estimate sitting posture from body pressure distribution images. The characteristics of the pressure distribution data of the sitting plate body were analyzed and pre-processed for deep-learning. Based on this, a deep-learning model for estimating sitting posture from body pressure distribution images was developed and verified. The classification performance of logistic regression and deep-learning model is compared.

Chapter 4 uses deep-learning to measure musculoskeletal disorders. The motion of factory workers was analyzed to extract motion through pre-processing. Deep-learning models have been developed for estimating OWAS, RULA, and REBA scores from the extracted motions. These were verified by test data and compared with each other.

Chapter 5 uses deep-learning to estimate the stature from the size information of a portion of the hand. The characteristics of the size information of a part of the hand were analyzed. RNN and RGLM models have been developed to estimate the stature from the size of a hand. And the performance of these models and linear regression model is compared.

Chapter 6 summarizes the findings of this study, suggests the contribution of this study, identifies the limitations of this study, and suggests further studies.

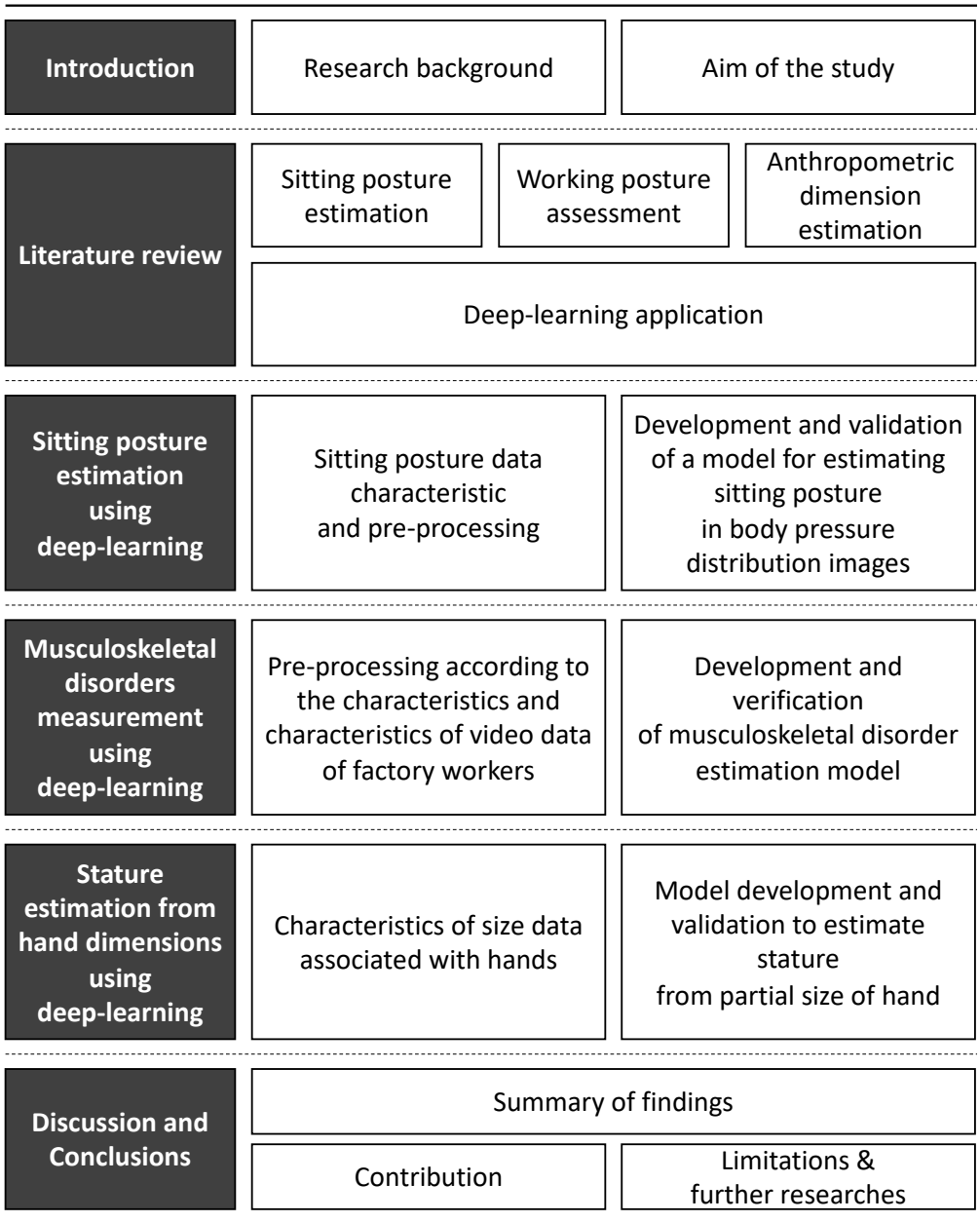


Figure 1.1 Organization of the thesis

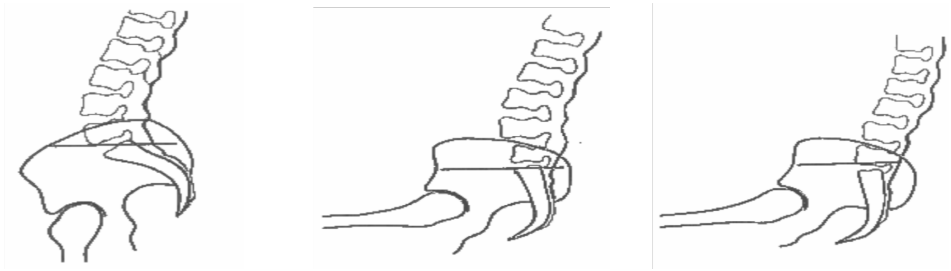
Chapter 2

Literature Reviews

2.1 Sitting Posture

Sitting posture is an ordinary physical posture. Many of us spend most of our day sitting at work, at home, driving, and out. And most people try to sit rather than stand. Sitting posture has been found to require less muscle work than standing posture (Andersson and Ortengren, 1974). In this study, they compared the standing posture, sitting posture unsupported, back rest inclination, seat inclination, lumbar support, and thoracic support. A decrease in pressure was obtained by an increase in backrest inclination and an increase in lumbar support.

As shown in Figure 2.1, sitting flattens the lumbar spine. As we sit, our hamstring muscles stretch, rotating our pelvis back. About 2/3 of this shift towards sitting is by flattening the lumbar spine with the rest from tilting of pelvis (Bendix and Biering-SÃžrensen, 1983). This flattens the lumbar curve excessively. On the other hand, standing causes the pelvis to rotate forward,



(a) Standing **(b)** Sitting Erect **(c)** Sitting Relaxed

Figure 2.1 The appearance of lumbar spine according to posture

thereby excessively increasing the lumbar curve depth.

However, continuous sitting has disadvantages and potential long-term consequences. Dunstan et al. (2010) found that cardiovascular disease mortality tended to increase significantly as the time to sit and watch television increased. Videman et al. (1990) noted that symmetric disc degeneration was associated with sedentary work. These results were obtained by multivariate analysis to determine the relationship between history of back pain and occupational characteristics for 86 male cadavers. Figure 2.2 shows the estimated marginal means for the associations of quartiles of breaks in sedentary time with waist circumference, and BMI. Compared to those in the lowest quartile of breaks in sedentary time, those in the highest quartile had, on average, a 5.95 cm lower waist circumference (P $\bar{0}$.025) (Healy et al., 2008).

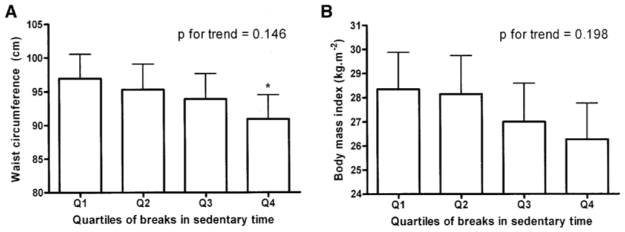


Figure 2.2 Quartiles of breaks in sedentary time with metabolic risk variables: waist circumference, BMI. (Healy et al., 2008)

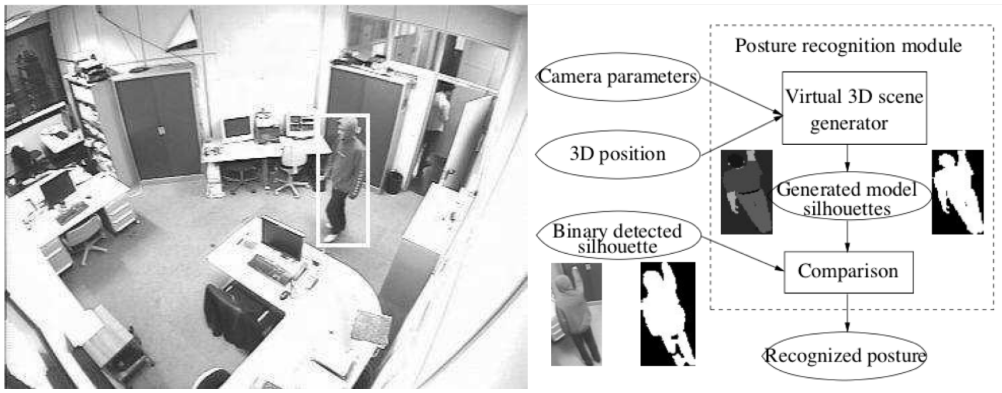
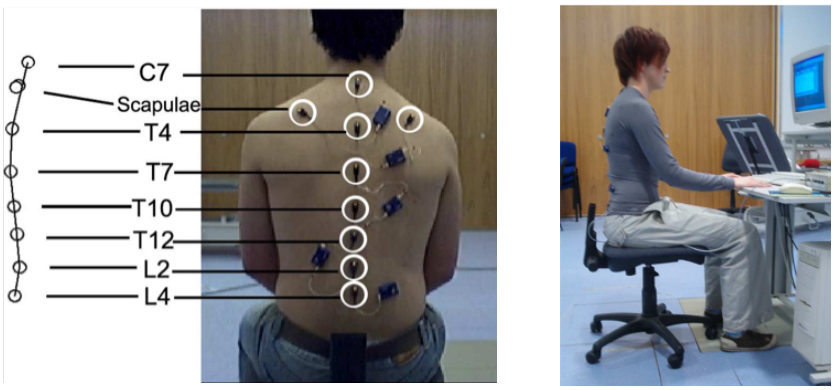


Figure 2.3 A procedure to recognize the actual person's posture by comparing the figure of the person who was photographed with the camera and the 3D model (Boulay et al., 2005)

Ergonomists tried to solve these problems by measuring and analyzing the sitting posture in various ways. The camera was used to measure and capture the sitting posture(Boulay et al., 2005). As shown in the Figure 2.3, the silhouette was extracted by taking a picture of a person using a camera and compared with the silhouette of a person who created it using a virtual 3D scene generator. As a classification technique, a region model that recognizes and compares regions of an image was used. General postures such as standing posture(standing with one arm up, standing with arms along with the body and T-shape posture.), sitting postures(sitting on a chair and sitting on the floor), bending postures, and lying postures(lying with spread legs and lying with curled up legs.) were classified with more than 95% accuracy.

Sensors can also be attached to the user's body to recognize the behavior of motion data and sitting postures. In Foerster et al. (1999)'s study, four piezoresistive sensors were attached to the sternum, wrist, thigh, and lower legs to classify lying, sitting, sitting/talking, sitting/operating, stainding, walking, walking/downstairs, walking/upstairs and cycling. The number of measurements in each posture varies from 2 to 210, but the accuracy is 0.667.



(a) Infrared markers **(b)** Garment with sensors

Figure 2.4 Using the infrared marker attached at the position of (a), the position of the sensor suitable for posture monitoring was derived and the garment was made as shown in (b). (Dunne et al., 2008)

In Dunne et al. (2008)'s study, the infrared marker was installed in 9 spines shown in Figure 2.4 (C7, L Scapula, R Scapula, T4, T7, T10, T12, L2, and L4) and the 3D motion capture system was used to analyze the spinal posture, and C7, T7, and L4 were regarded as the optimal sensor positions for posture measurement. Garment with fiber-optic bend sensing was proposed as a wearable monitoring of seated spinal posture.

Although there are differences in the position and number of sensors attached, most studies were performed by measuring and classifying the posture by attaching pressure sensors directly to the chair. Tan et al. (2001) classified 14 postures by measuring body pressure distribution using a 42 by 48 pressure sensing sheet attached to the seat and back. As a result of classifying body pressure distribution using PCA, the accuracy of the users (30 people) included in the training data was 96%. The accuracy of the users (8 people) not included in the training data was 79%. As shown in Figure 2.5, there is a big difference in accuracy, depending on whether the subject is familiar with it. Mota and

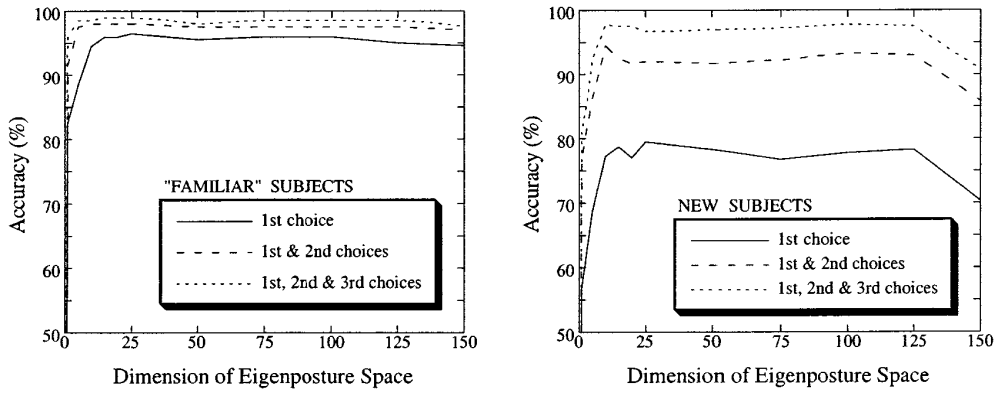


Figure 2.5 Classification accuracy for "familiar" and new subjects. Solid, dashed and dotted lines correspond to accuracy associated with the smallest ε value, first two smallest ε values, and the first three smallest ε values, respectively (Tan et al., 2001).

Picard (2003) measured nine postures for ten children aged 8 to 11 years and classified them using HMM. The accuracy was 82.2% when the test subject's information was included, and 76.4% when it was not included.

Table 2.1 shows the previous studies that performed posture prediction including the number of sensors, participant information, and predicted postures. In the case of the sensor configuration, when only the sensor in the seat was used, the sensor was installed in the seat and backrest, and the sensor was installed in the seat, backrest, and armrest. In three cases, the posture was recognized using the sensor installed on the seat. The number of sensors used varied from 3 to 4032. The number of posture classification subjects ranged from 5 to 52. There were cases in which gender was distinguished and in cases where gender was not distinguished. In many cases, information about the age of the subjects was not provided. In the case of age information provided, most of the subjects were adults. The number of postures classified in each study ranged from 4 to 14. The accuracy derived from each study ranged from 0.56 to 0.995.

Table 2.1 Previous studies that predicts sitting postures

Sensor location	Number of sensors	Number of participants (Total, Male, Female)	Age information (yrs)	Number of Detection conditions	Accuracy	Previous studies
Seat	5	10, none, none	No information	8	0.938	Bao et al. (2013)
Seat and backrest	Seatpan (8) Backrest (8)	9, none, none	32-45	12	none	Barba et al. (2015)
Seat and backrest	Kinetic sensor (3) Temperature (1) Pressure (5)	7, none, none	No information	6	0.937	Benocci et al. (2011)
Seat	64	5, none, none	No information	4	none	Fard et al. (2013)
Seat and backrest	Seat (2) Backrest (4)	No information	No information	6	none	Hu et al. (2010)
Seat	64	10, 10, 0	21-24	9	0.906	Kamiya et al. (2008)
Seat and backrest	Seat (6x8) Backrest (2x8)	7, none, none	20-31	9	0.823	Xu et al. (2012)
Seat and backrest	Seat (2) Backrest (1)	20, 12, 8	No information	6	0.995	Ma et al. (2016)
Seat	textile sensor with 240 sensor elements	9, 6, 3	No information	16	0.56(only seat sensor) 0.84(with back sensor)	Meyer et al. (2010)
Seat and backrest	Seat (42x48) Backrest (42x48)	10, 5, 5	8-11	9	0.76	Mota and Picard (2003)
Seat and backrest	**1st experiment:** Seat (42x48) Backrest (42x48)	**1st experiment:** 52, 26, 26	**1st experiment:** No information	10	0.87	Mutlu et al. (2007)
	2nd experiment Seat (11) Backrest (8)	**2nd experiment:** 20, 10, 10	**2nd experiment:** 19-34			
Seat and backrest	Seat (42x48) Backrest (42x48)	**For single user:** 1 **For multiuser:** 30, 15, 15	**For single user:** No information **For multiuser:**, 18 – 60	**For Single usre:** 14 **For multiuser:** 10	0.96(familiar) 0.79(non-familiar)	Tan et al. (2001)
Seat	eTextile (made of fibers coated with conductive polymer)	25, 15, 10	No information	7	0.859	Xu et al. (2013)
Seat, backrest and armrest	Seat (10) Backrest (4) Armrest (3x2 arms)	41, 25, 16	24–64	7	0.876	Zemp et al. (2016)

2.2 Working Posture Measurement

In general, posture means that the positions of all the joints of the body at a given moment are synthesized (Dendamrongvit, 2002). Posture is defined in various ways, taking into account biomechanical alignment, the spatial arrangement of body parts, the relative position between segments, and the body posture that is considered to perform the task (Haslegrave, 1994; Rohmert and Mainzer, 1986).

Working posture and movement are one of the important mechanical and load determinants, so they are important variables to consider for industrial safety. Working posture was assessed subjectively using various observational checklists such as the Ovaco Working Postures Assessment System (OWAS), the Rapid Upper Limb Assessment (RULA), and the Rapid Entire Body Assessment (REBA).

OWAS is an evaluation technique developed by the Ovako Steel Company to define and evaluate workers' inappropriate working postures (Karhu et al., 1977). It is mainly suited for the handling of heavy loads by personnel, and scores from 1 to 4 points. The first category related to normal postures without recommendations of any type for corrective activity. The second and third categories concerned postures with some risk with recommendations for corrective actions to be taken over the middle term. The fourth category referred to unacceptable postures with recommendations for immediate corrective measures.

The OWAS method was intended to identify the frequency and time spent in the postures adopted in a given task, to study and evaluate the situation, and thus, recommends corrective actions. OWAS observes the most habitable postures of the back, arm, and leg that the worker takes. As shown in Figure 2.6,

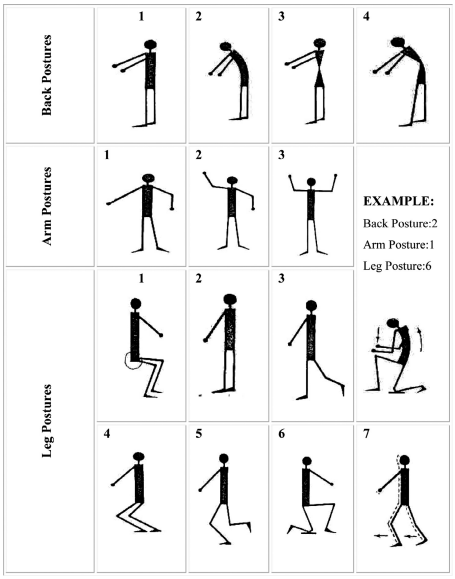


Figure 2.6 Standard working postures in the OWAS method (Luopajarvi, 1990)

Table 2.2 Action categories(AC) in OWAS from Karhu et al. (1977)

	back	1			2			3			4		
leg	a \ b	1	2	3	1	2	3	1	2	3	1	2	3
1	1	1	1	1	2	2	3	1	2	2	2	3	4
	2	1	1	1	2	2	3	1	2	2	3	3	4
	3	1	1	1	3	3	4	1	3	3	3	4	4
2	1	1	1	1	2	2	2	1	1	1	2	2	2
	2	1	1	1	2	2	2	1	1	1	2	3	3
	3	1	1	1	3	3	3	1	1	1	3	4	4
3	1	1	1	1	2	2	3	1	1	2	2	3	3
	2	1	1	1	2	3	3	1	1	3	2	3	3
	3	1	1	1	3	3	3	2	2	3	3	4	4
4	1	2	2	2	3	3	3	3	4	4	4	4	4
	2	2	2	2	3	4	4	3	4	4	4	4	4
	3	2	2	3	3	4	4	3	4	4	4	4	4
5	1	2	2	2	3	3	4	4	4	4	4	4	4
	2	2	2	2	3	4	4	4	4	4	4	4	4
	3	2	2	3	3	4	4	4	4	4	4	4	4
6	1	1	1	1	2	3	4	1	3	4	4	4	4
	2	1	1	1	2	3	4	1	3	4	4	4	4
	3	1	1	1	2	4	4	1	3	4	4	4	4
7	1	1	1	1	2	2	2	1	1	1	2	2	2
	2	1	1	1	3	3	3	1	1	1	3	3	3
	3	1	1	2	3	4	4	1	1	1	4	4	4

a: load, b: arm

there are four back postures, three arm postures, and seven leg postures. Also, the weight of the weight handled by workers is observed in 3 categories. These make a total of 252 possible combinations. Therefore, each posture assumed by a worker was assigned a 4-digit code that depended on the classification within the previous postures for each part of the body and the load.

The procedure for applying OWAS is as follows. It consists of observing work tasks, coding postures according to Figure 2.6, deriving action categories according to Table 2.2, and suggesting corrective postures.

RULA is an evaluation technique developed by the University of Nottingham in the UK to prevent occupational upper extremity disease (McAtamney and Corlett, 1993). RULA is suitable for the site where there is a lot of upper limb work, such as an automobile assembly line, and it is evaluated by a score from 1 to 7 points.

RULA was developed to quickly and efficiently evaluate the workload due to the working tax by focusing on the upper limbs, such as the shoulder, wrist, wrist, and neck. RULA's purpose is to provide a quick and easy way to determine the percentage of workers with upper limb disorders due to bad working posture. Also, it evaluates the muscle load by factors such as working posture, static or repetitive work, and strength, which are factors affecting muscle fatigue.

The values for the degree of use, frequency of use of the arms, wrists, neck, torso, legs, and muscles are evaluated using the table in Figure 2.7. This method mainly evaluates the risk of working posture quantitatively and performs post-management according to the final evaluation score. 1 to 2 points are appropriate tasks, 3 to 4 points are tasks requiring follow-up, 5 to 6 points are tasks that need to be considered, and 7 points require immediate task change.

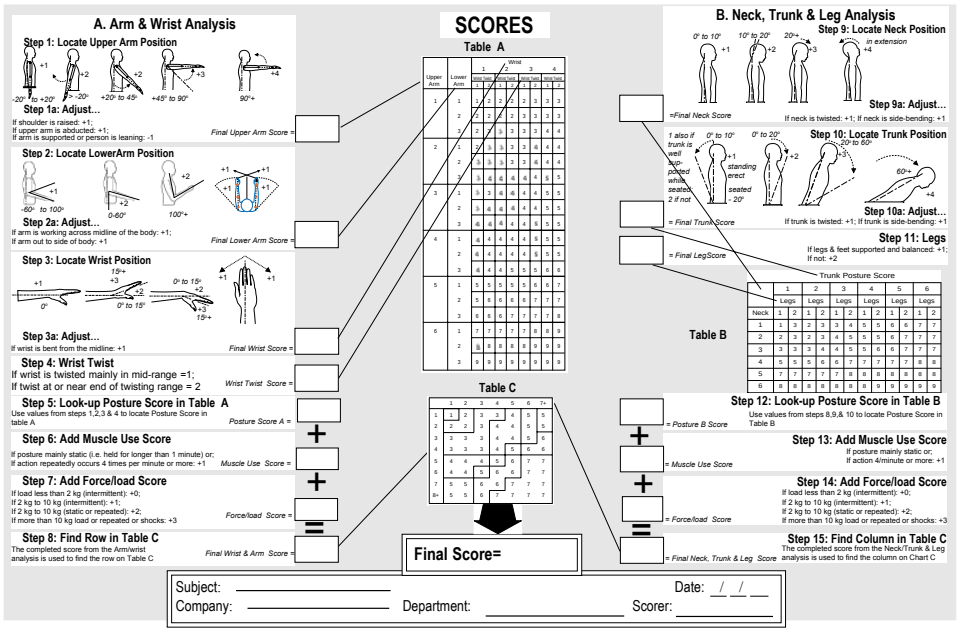


Figure 2.7 RULA employee assessment worksheet (Hedge, 2000b)

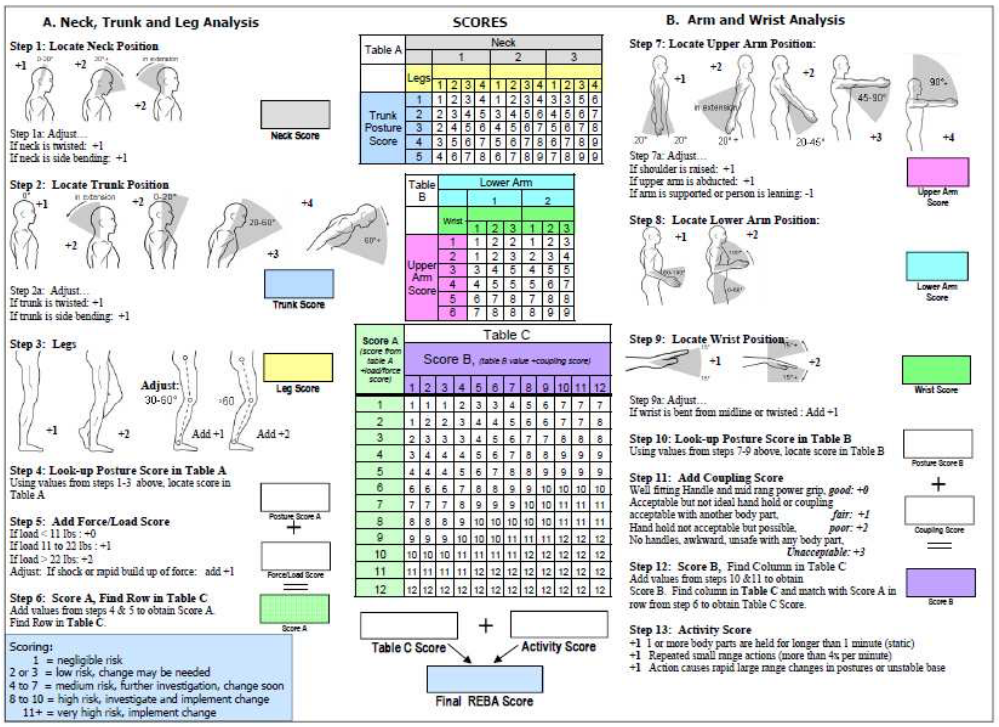


Figure 2.8 REBA employee assessment worksheet (Hedge, 2000a)

REBA is a working posture analysis tool developed to be sensitive to unpredictable work posture (McAtamney and Hignett, 2000). REBA observes the worker's movement stage, divides body parts, and assigns points to each body part.

REBA was developed to evaluate the degree of exposure of individual workers to risk factors related to musculoskeletal disorders. Compared to RULA, which mainly evaluates upper limb work, it is suitable for analyzing cases of working in various postures that are difficult to predict.

The REBA consists of four scorecards representing work posture by body parts, as shown in Figure 2.8. The main work factors to be evaluated are repeatability, static work, force, working posture, and continuous work time. As-

sessment is based on the body part, working posture, muscles, and strength.

The evaluation result is expressed as a total score between 1 and 15 points. It is classified into five action levels according to the score. Action level 0 means no particular action is required. Action level 1 may require action. Action level 2 requires action; action level 3 requires action soon. Action level 4 means that immediate action is required.

Since these evaluation tools rely on the evaluator's observations, they have the following limitations: subjectivity, evaluator's bias, low precision, long analysis period, and highly trained observer's need (Yen and Radwin, 2002).

Unlike these methods, the measurement of working posture and movement through quantitative biomechanical measures was more accurate and reliable than using the checklist (Juul-Kristensen et al., 2001). The use of such measuring equipment has the disadvantage of affecting or disturbing the work and distorting the work characteristics.

2.3 Anthropometric Dimension Estimation

Among the body parts humans have, the hand is the body part that allows the most delicate movement and allows various motions. The hand is composed of a lot of muscles and joints. Moreover, it takes different shapes for different people (Chaffin et al., 2006).

The size information of hands is used not only in the development of products and interface design, but also in confirming the identity of presumed personnel in forensic and emergency medical science (Kanchan et al., 2012). For example, when conducting an analysis of the traces remaining on the scene of the incident, the length of the handprints are used for the estimation of gender

or height (Kanchan and Rastogi, 2009). In crime scenes, there are many cases where the only evidence that presumes a criminal is a handprint (Kanchan et al., 2012). So, the Forensic anthropometry method using hand size is a useful tool for confirming the presumed personnel's identity.

The existing research estimated gender, length, or height using the measured data of various parts of hands. Hand anthropometry measures various size information of hands, such as the information on sizes and shapes. It uses the hand anthropometry information on the product development and establishment of the system. The existing researches recognize the importance of the information of hands, and various hand anthropometry researches were conducted (Hadler et al., 1978). Since the hand is the most frequently used body part, traces related to the hand often remain on the crime scene. Therefore, from the past, hand anthropometry has been used to determine the identity of criminals using hand-related measurement information (Aboul-Hagag et al., 2011).

Several previous studies used parts of the hand to estimate stature. Agnihotri used the hand lengths and breadths of 250 Mauritius students to predict their heights (Agnihotri et al., 2008). In this study, hand length was the most critical factor in estimating stature. A study conducted on Egyptians showed a high correlation between phalange lengths and stature (Habib and Kamal, 2010). In Ahmed's study of 503 Indian male students, hand length best estimate stature (Ahmed, 2013). A study has derived a regression equation that estimates the stature with high accuracy from the upper limb and hand (Akhlaghi et al., 2012). Ahmed's study used several parts of the body to estimate the stature and found that hand length was more predictable than handbreadth (Ahmed, 2013). Uhrová's study did not show a difference between the right hand and

the left hand in estimating the stature (Uhrová et al., 2013).

In Jee and Yun (2015), a variable combination was derived by stepwise regression analysis to estimate the stature using the size of each part of the hand. The hand size combinations for estimating the stature of both men and women were WC, PL, 3DL, and 3D3L, with an R2 value of 0.642 with an estimated error of 5.719 cm. Hand size combinations for estimating male stature were HL, 3DM2L, and PL, with an R2 value of 0.425 with an estimated error of 4.819 cm. Hand size combinations for estimating female stature were HL, MHB, 3DM1L, and 1DT1L, with an R2 value of 0.418 with an estimated error of 5.080 cm.

Brolin et al. (2017) proposed an adaptive regression model for synthesizing anthropometric population data. Although it showed valid results, it showed a problem that the accuracy did not increase even if the data size increased.

2.4 Deep-learning Application

LeNet-5 is the CNN architecture used for handwritten digit recognition in the early days and successfully used to solve other visual related problems (Lin et al., 2016; Sarvadevabhatla and Babu, 2015). As shown in Figure 2.9, the LeNet-5 model consists of an input, three convolution layers, two sub-sampling layers, a fully-connected layer, and an output layer. The C1 layer performs convolution operations on 32 x 32 images to obtain six 28 x 28 feature maps. In the S2 layer, subsampling is performed on 6 28 x 28 feature maps and reduced to 14 x 14 feature maps. The C3 layer extracts 16 10 x 10 feature maps by performing a convolution operation on 6 14 x 14 feature maps. In the S4 layer, subsampling is performed on 16 10 x 10 feature maps and reduced to 16 5 x 5 feature maps. In the C5 layer, 120 1 x 1 feature maps are calculated by filtering

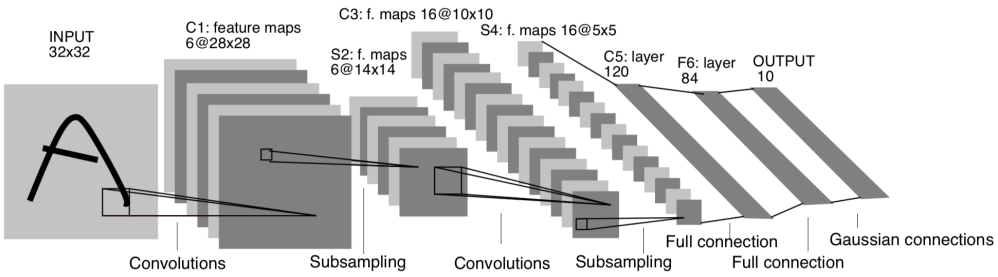


Figure 2.9 Layer structure of LeNet-5(LeCun et al., 1998)

16 5 x 5 feature maps. The F6 layer links the results of C5 to 84 units. The output layer tells the class the image belongs to.

LSTMs are a special kind of RNN, capable of learning long-term dependencies and remembering information for prolonged periods as a default. LSTM is a special kind of RNN that can learn long-term dependencies and remember long-term information. the LSTM model consists of a chain structure. However, the structure of the repeat module is different. Four layers interact with unique communication methods instead of a single neural network like standard RNN (Gers et al., 1999). Le et al. (2019) also predicted the flow of water to 99% NSE as seen in the figure 2.10 using LSTM.

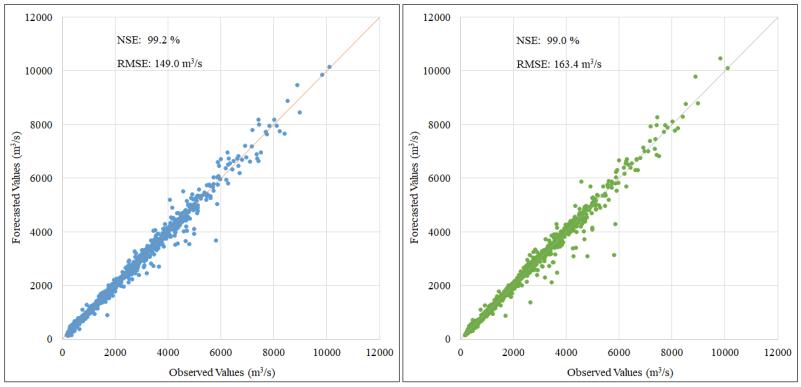


Figure 2.10 Scatter plot for one-day flow forecasting (Le et al., 2019).

Chapter 3

An Ergonomic Analysis of Seated Posture using a Deep-learning Method

3.1 Overview

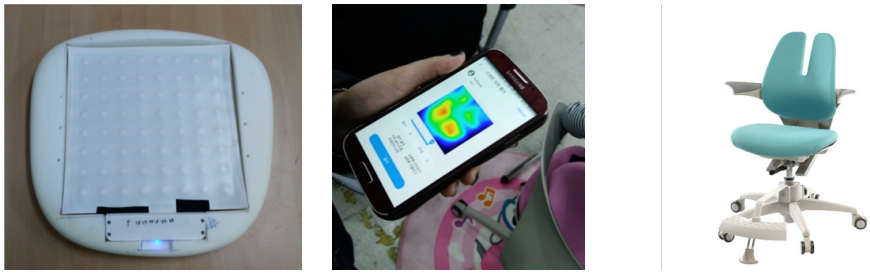
Sitting position is a factor that affects back health(O'sullivan et al., 2002). Posture is a habit, so it is crucial for a child to have a good posture. When the user sits in a chair, the body pressure distribution varies according to the posture. This body pressure distribution can estimate the user's posture. This posture estimation is a classification problem and can be applied to machine learning or deep-learning. In this chapter, the body pressure distribution, according to the user's posture, is classified through deep-learning.

3.2 Data Characteristics

3.2.1 Body Pressure Distribution on Seat Cushion

The body pressure distribution data of the chair cushion was used to classify the sitting posture. The body pressure sensor should be used to measure the body pressure of the chair cushion. At this time, studies using a system for measurement of body pressure consisted with more than 1000 pressure sensors such as Tekscan's CONFORMAT System has been conducted(Mota and Picard, 2003; Tan et al., 2001). However, it has disadvantages that the equipment is expensive and can be used only in a laboratory environment. In order to measure the pressure distribution even outside of the laboratory environment, there have been studies using 10 pressure sensors attached to the chairs(Bao et al., 2013; Barba et al., 2015; Ma et al., 2016; Zemp et al., 2016). In these cases, there are advantages that the cost is low and the sensor can be used in a non-lab environment instead of a laboratory environment. However, there is a disadvantage in that the amount of information that can be measured is small as the number of sensors is small.

In this study, 8x8 pressure sensors were placed on the film as shown in Figure 3.1 (a). This is a prototype product developed by DuoBack and Algorigo to be launched as a consumer product. Therefore, data for training and testing were collected using products that are likely to be released as actual products, rather than a pressure sensor study that was performed only in a laboratory environment. In addition, the chair used was RA-070SDSF from Duoback, which was made for children in Figure 3.1 (c). The chair can be adjusted so that it does not rotate, and the footrest can be adjusted to suit the body shape of the subjects. pressure sensors were installed inside the seat cushion of a child



(a) (b) (c)

Figure 3.1 The component of data collection system for the experiment. (a) Chair cushion with pressure sensor, (b) android smartphone environment for receiving pressure data, (c) Duoback RA-070SDSF

chair. Pressure sensors with a diameter of 15 mm were arranged at intervals of 36 mm. The pressure measuring area is 318mm × 318mm, and the PET film sensor has a force sensing resister (FSR) of 8×8. The data is transferred to android smartphone via Bluetooth network with 12bit (Figure 3.1 (b)). The sensor data was recorded at a cycle of 2 Hz.

3.2.2 Data Collection

Seven sitting postures were selected for data collection(Figure 3.2). Based on the results of previous studies on posture prediction, the representative postures were selected including sitting upright, lean forward, lean left, lean right, lean backward. In addition, to contain any awkward postures reflecting children's sitting behavior, pre-online interview was conducted for the parents. A total of 32 parents completed questionnaire on their child's unusual sitting postures. As a result, two more sitting postures were contained in this experiment; one is sitting at the edge of seat pan and the other is sitting posture in which the knees are bent and crossed inward and both feet are placed under the opposite leg knee.

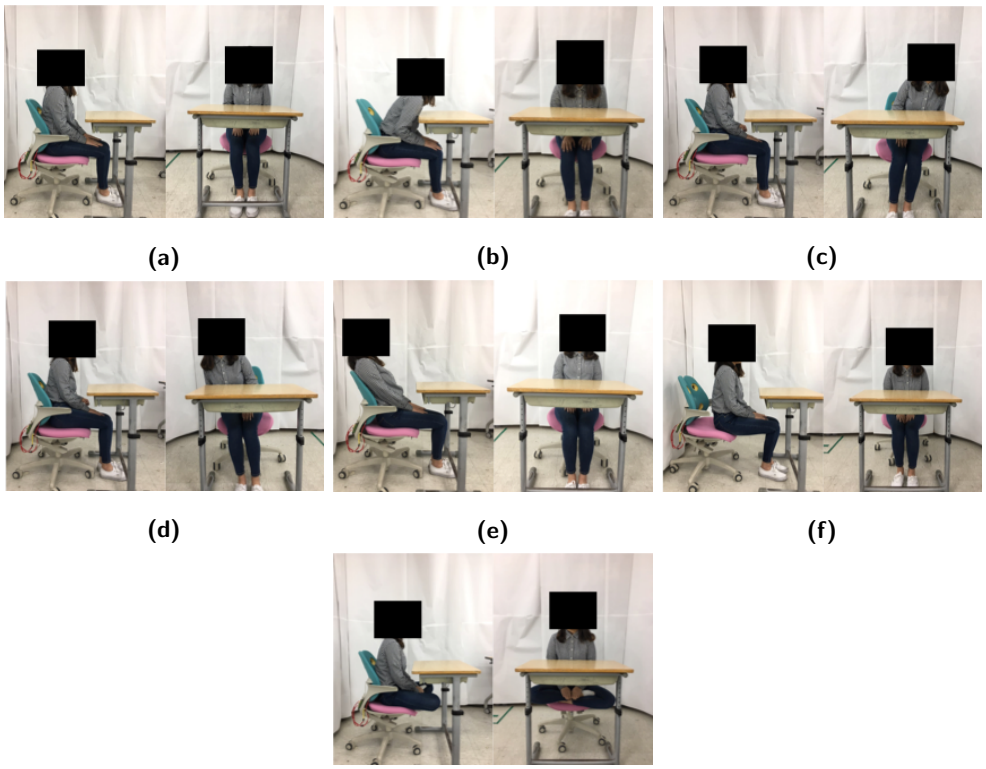


Figure 3.2 Seven sitting postures used in data collection. (a) Sitting straight, (b) Lean forward, (c) Lean left, (d) Lean right, (e) Lean backward, (f) Sitting at the front of the chair, (g) Sitting crossed-legged on the chair

Data collection was conducted in an environment similar to the actual chair use environment. The subject's main task is to sit down and write. The height of the chair was adjusted so that the feet reached the floor and the thighs did not fall off the seat. In order to prevent the posture from being deformed according to the posture placement order, different posture placement procedures were applied to each participant.

Body pressure distribution data were collected at a frequency of 2 Hz. Body pressure distribution data was recorded for 32 seconds after the first data recording, and 65 body pressure data were recorded per session. After obtaining 65 data, one minute of rest was given to subject, and then measured again. As a result of 4 data collection sessions for one posture, 260 pressure data for one posture could be collected. After data on one posture were collected, the subjects were allowed to take a five-minute break and collect data on the other postures. The body pressure distribution data collection for the seven postures of the subjects was performed for about 66 minutes, and a total of 2080 body pressure distribution data were collected for one subject.

3.2.3 Data Pre-processing

The pressure data collected by the pressure sensor on the chair seat was stored in an 8 by 8 array. Manual calibration was performed by the operator during the individual measurements, but scaling of the measured data is necessary. If the value of the data is too big or too small, it may converge to zero or diverge indefinitely during model training, so data scaling should be preprocessed. Scaling can be done by simple code insertion during training and validation, but the reason for applying it in the pre-processing is to facilitate image analysis and result interpretation.

LeNet-5, a deep-learning model for image classification, was used to classify the posture with pressure distribution data. Since input data of LeNet-5 is image, sensor data should be converted into image.

The most suitable form of visualization of pressure distribution data in an 8 by 8 array is a heat map. Since data scaling is required, the color spectrum of the heat map is grayscale. This is a MinMaxScaler with 0 as white and 1 as black.

The heat map conversion of pressure sensor values was done using R, a programming language for statistical calculations and graphics processing. Plotly, an open source graphing library, was used for heat map generation. There are three interpolation methods that can be selected when generating a heat map in Plotly: best, fast, and false. Among them, best, a bilinear interoplation method, is selected. The generated heatmap was saved in png graphic format at the size of 1072 x 824.

The LeNet-5 is not a good model for processing 1072 x 824 images, so I converted the image size to 100 x 77. Since the LeNet-5's input data is square in shape, the image size is adjusted to 100 by 100 by padding the top and bottom of the resized heatmap.

An example of sensor data pre-processing is shown in Figure 3.3. (a) Pressure distribution data is collected through 8 by 9 sensors on the seat. (b) Create a heatmap with the minimum value of the collected data as white and the maximum value as black. (c) The image is converted by applying interpolation to the heatmap. (d) Convert the image size according to the CNN model to be applied.

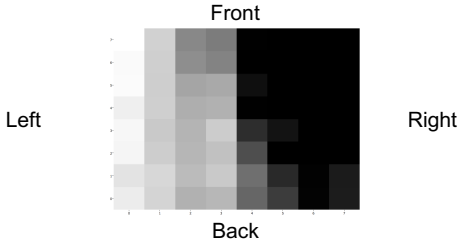
Front

64	413	977	1068	2018	2024	2024	2020
100	426	934	1014	2022	2025	2025	2025
91	439	755	720	1909	2026	2025	2025
184	429	683	664	2015	2025	2026	2025
121	476	635	452	1672	1875	2023	2023
138	449	622	534	1421	2024	2023	2019
274	365	581	478	1169	1707	2005	1819
202	388	661	613	1238	1561	2001	1806

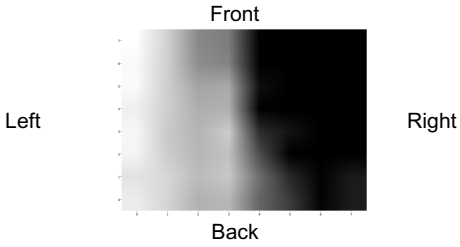
Left Right

Back

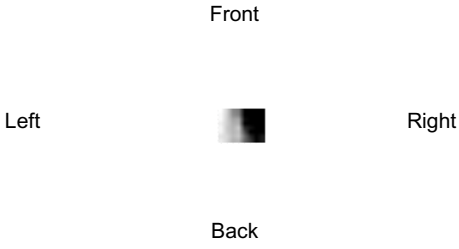
(a) Pressure sensor data



(b) Heatmap without interpolation



(c) Heatmap with interpolation



(d) Reduced size heatmap

Figure 3.3 Example of sensor data preprocecing. (The size of the rectangle in (d) reflects the actual size of data reduction.)

3.3 Data Analysis

Among the many deep-learning methods, CNN was chosen because it is suitable for image classification. The algorithms that correspond to the CNN method include LeNet-5, AlexNet, VGGNet16, Google Net, Inception, and ResNets. The most simple structure among these algorithms, the LeNet-5 was chosen.

The deep-learning model shows a difference in performance depending on the composition of the training data. As a result, one model cannot correctly evaluate the performance of deep-learning applied to a model. Cross-validation is applied to reduce the performance difference according to the structure of the training data. Cross validation was performed on 29 data from 32 data. The 29 data also included three outliers to analyze the impact of individual user differences. The overall performance and classification characteristics of the 29 sitting posture classification models were analyzed.

Besides, logistic regression was performed to compare deep learning performance with existing classification techniques. As in deep learning, cross-validation was applied to logistic regression.

3.3.1 Convolutional Neural Network

LeNet-5 is the CNN architecture used for handwritten digit recognition in the early days and successfully used to solve other visual related problems (Lin et al., 2016; Sarvadevabhatla and Babu, 2015).

LeNet-5 model Modification

Since the LeNet-5 model is a model used for small-sized datasets, some modifications have been made to this study. Because the size of the image has increased and the number of images to be classified has changed, the model needs to be modified. The input layer has been modified to accommodate the size of the image, and the two fully-connected layer has been modified accordingly.

Layer Structure

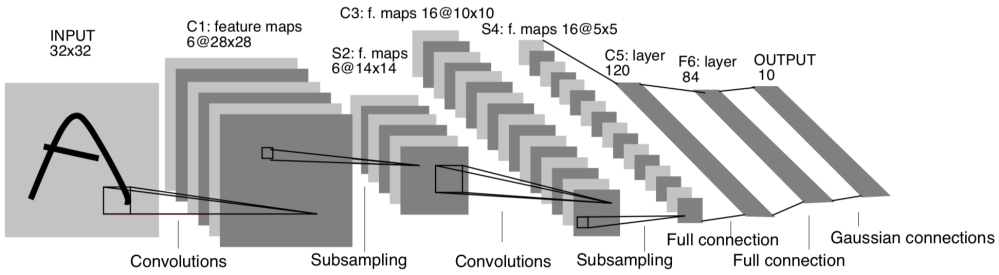


Figure 3.4 Layer structure of LeNet-5(LeCun et al., 1998)

As shown in Figure 3.4, the LeNet-5 model consists of an input, three convolution layers, two sub-sampling layers, a fully-connected layer, and an output layer. The C1 layer performs convolution operations on 32 x 32 images to obtain six 28 x 28 feature maps. In the S2 layer, subsampling is performed on 6 28 x 28 feature maps and reduced to 14 x 14 feature maps. The C3 layer extracts 16 10 x 10 feature maps by performing a convolution operation on 6 14 x 14 feature maps. In the S4 layer, subsampling is performed on 16 10 x 10 feature maps and reduced to 16 5 x 5 feature maps. In the C5 layer, 120 1 x 1 feature maps are calculated by filtering 16 5 x 5 feature maps. The F6 layer links the results of C5 to 84 units. The output layer tells the class the image belongs to.

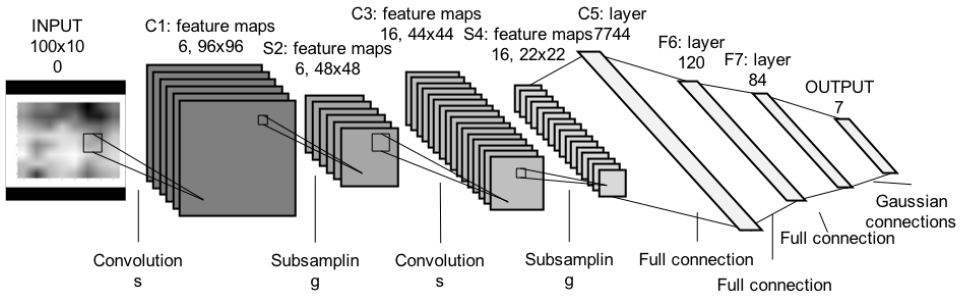


Figure 3.5 Layer structure of modified LeNet-5 model in this study

Since the size of the image used in this study is larger than that of the LeNet-5, the layer structure has been modified to handle this. As shown in Figure 3.5, the C1 layer performs convolution operations on 100 x 100 images to obtain six 96 x 96 feature maps.In the S2 layer, six 96 x 96 feature maps are reduced to six 48 x 48 feature maps. In the C3 layer, 16 44 x 44 feature maps are obtained by convolutional operations on 6 48 x 48 feature map. In the S4 layer, 16 44 x 44 feature maps are reduced to 16 22 x 22 feature maps. In the C5 layer, 7744 1 x 1 feature maps are generated by filtering 16 22 x 22 feature maps. In the F6 layer, 7744 1 x 1 feature maps are connected to 120 units. In the F7 layer, 120 units are connected to 84 units. The output layer tells the class the image belongs to.

3.3.2 Performance Comparison Method

LeNet-5 is a model for classifying images. The performance of the classification model can be assessed using the relationship between the actual values and the classification results. The actual value is divided into positive and negative, and the classification result is divided into positive and negative. These can be arranged in a 2 x 2 matrix as shown in Table 3.1. Precision is the ratio of what the model classifies as positive to actual positive(Equation (3.1)). Recall is the

Table 3.1 Confusion matrix for the relationship between actual values and classification results

		Actual Values	
		Positive	**Negative**
Predicted Values	**Positive**	True Positive(TP)	False Positive(FP)
	Negative	False Negative(FN)	True Negative(TN)

ratio of what the model classifies as positive to true(Equation (3.2)). The F1 score is the harmonic mean of precision and recall(Equation (3.3)). Accuracy is an index indicating the ratio of positive classification to positive and negative classification to negative(Equation (3.4)).

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive} \tag{3.1}$$

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative} \tag{3.2}$$

$$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{3.3}$$

$$Accuracy = \frac{TruePositive + TrueNegative}{TruePositive + TrueNegative + FalsePositive + FalseNegative} \tag{3.4}$$

3.4 Results

3.4.1 Logistic Regression

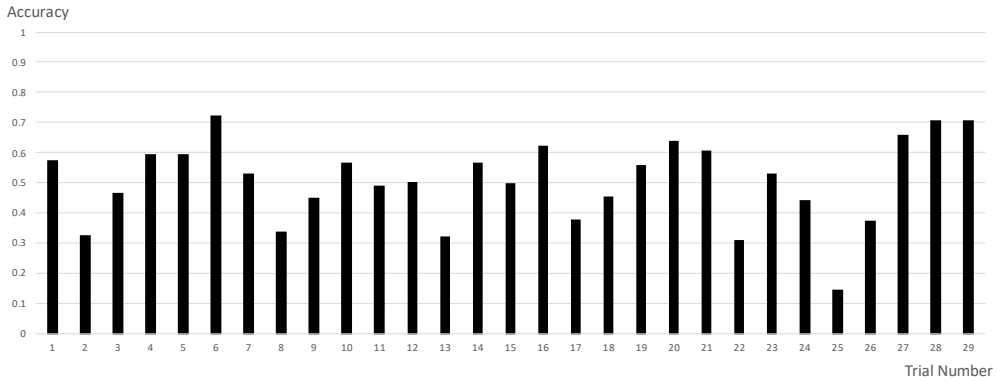


Figure 3.6 Accuracy value according to cross-validation applying Logistic regression

The distribution of accuracy can be summarized, as shown in Figure 3.6. A total of 29 cross-validations were performed. Of those, 17 accuracy was more significant than 0.5, and 12 accuracy was less than 0.5. Accuracy was best when the 6th participant's data was used as the test data, and the rest was used as training data. When the 25th participant's data was used as the test data, and the rest was used as training data, the accuracy was worst.

The overall results of each test are summarized with accuracy and F1 score. The results of the cross-validation of 29 pressure distribution data using logistic regression for each data are shown in Table 3.2. Accuracy is 0.148 for the minimum value and 0.723 for the maximum value. The average value is 0.507(SD=0.138). When sitting straight (a) is classified, the average F1 score is 0.244(SD=0.291). When lean forward (b) is classified, the average F1 score is 0.169(SD=0.196). When classifying Lean left (c), the average F1 score is 0.664(SD=0.278). When lean right (d) is classified, the average F1 score is

0.637(SD=0.238). When lean backward (e) is classified, the average F1 score is 0.125(SD=0.206). When sitting at the front of the chair (f), the mean F1 score is 0.620(SD=0.359). When sitting sitting-legged on the chair (g), the mean F1 score is 0.700(SD=0.289).

Table 3.2 F1 score and accuracy of cross-validation results on logistics regression

	Accuracy	F1 Score						
		a	b	c	d	e	f	g
1	0.574	0.221	0.336	0.604	0.921	0.633	0.960	0.506
2	0.327	0.000	0.000	0.821	0.331	0.000	0.306	0.166
3	0.466	0.000	0.360	0.382	0.853	0.143	0.000	1.000
4	0.596	0.638	0.000	0.677	0.629	0.000	0.613	0.970
5	0.593	0.000	0.000	0.829	0.717	0.454	0.992	0.757
6	0.723	0.000	0.489	0.962	0.729	0.587	1.000	0.968
7	0.531	0.000	0.000	0.987	0.571	0.380	0.805	0.791
8	0.339	0.361	0.296	0.541	0.420	0.000	0.008	0.649
9	0.453	0.000	0.348	0.917	0.497	0.000	0.295	0.816
10	0.569	0.427	0.000	0.631	0.830	0.000	0.996	0.644
11	0.493	0.000	0.000	0.916	0.440	0.518	0.466	0.634
12	0.505	0.592	0.209	0.903	0.739	0.000	0.015	0.763
13	0.323	0.000	0.064	0.029	0.456	0.000	0.691	0.431
14	0.568	0.000	0.000	0.962	0.425	0.000	0.886	0.998
15	0.499	0.015	0.364	0.725	0.639	0.000	0.621	0.998
16	0.622	0.520	0.074	0.732	0.625	0.000	0.927	0.906
17	0.380	0.000	0.323	0.537	0.527	0.000	0.765	0.158
18	0.454	0.456	0.016	0.716	0.578	0.035	0.527	0.834
19	0.559	0.492	0.081	0.821	0.761	0.000	0.845	0.424
20	0.638	0.348	0.306	0.737	0.875	0.000	0.998	0.958
21	0.606	0.614	0.000	0.672	0.797	0.000	0.850	0.647
22	0.310	0.000	0.428	0.031	0.561	0.317	0.210	0.274
23	0.533	0.389	0.000	0.815	0.863	0.000	0.943	0.452
24	0.445	0.000	0.097	0.656	0.123	0.365	0.735	0.954
25	0.148	0.257	0.000	0.000	0.000	0.000	0.000	0.000
26	0.377	0.000	0.000	0.371	0.821	0.107	0.000	0.851
27	0.661	0.907	0.000	0.511	0.883	0.000	0.787	0.994
28	0.709	0.850	0.468	0.773	0.983	0.008	0.768	1.000
29	0.709	0.000	0.647	0.987	0.870	0.067	0.972	0.755
Ave.	0.507	0.244	0.169	0.664	0.637	0.125	0.620	0.700
Std.	0.138	0.291	0.196	0.278	0.238	0.206	0.359	0.289
Min.	0.148	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Max.	0.723	0.907	0.647	0.987	0.983	0.633	1.000	1.000

Table 3.3 It is a confusion matrix that sums up the results of cross-validation using logistic regression.

Actual posture	Predicted posture								
	a	b	c	d	e	f	g	Total	Recall
a	2621	1907	493	791	1392	98	238	7540	0.348
b	1729	2258	919	948	1016	347	323	7540	0.299
c	458	903	5409	28	426	21	295	7540	0.717
d	626	626	40	5125	770	214	125	7526	0.681
e	1467	1691	732	960	1568	591	472	7481	0.210
f	275	1360	498	370	308	4593	136	7540	0.609
g	554	476	444	457	418	7	5184	7540	0.688
Total	7730	9221	8535	8679	5898	5871	6773	Accuracy	
Precision	0.339	0.245	0.634	0.591	0.266	0.782	0.765	0.508	

The overall results were summed up in the confusion matrix of Table 3.3. The accuracy value is 0.508. The recall value in (a) is 0.348, and the precision value is 0.339. The recall value in (b) is 0.299, and the precision value is 0.339. The recall value in (c) is 0.717, and the precision value is 0.634. The recall value in (d) is 0.681, and the precision value is 0.591. The recall value in (e) is 0.210, and the precision value is 0.266. The recall value in (f) is 0.609, and the precision value is 0.782. The recall value in (g) is 0.688, and the precision value is 0.765.

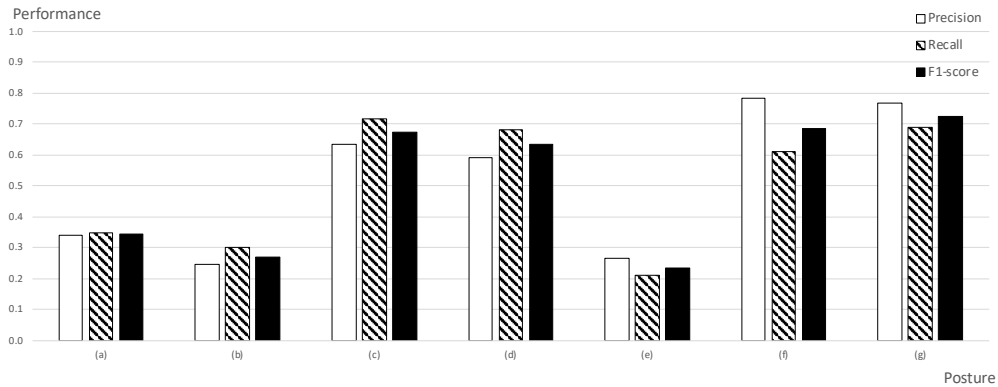


Figure 3.7 Average of precision, recall, and F1 score for each sitting posture. (a) Sitting straight, (b) Lean forward, (c) Lean left, (d) Lean right, (e) Lean backward, (f) Sitting at the front of the chair, (g) Sitting crossed-legged on the chair

The mean values of the precision, recall, F1 score of each sitting posture for posture classification results are shown Figure 3.7. The sitting precision of the posture of the sitting posture is the descending order of (f), (g), (c), (d), (a), (e), and (b). The precision value of sitting posture (f) is 0.782. In the case of a recall of a sitting posture, the descending order is (c), (g), (d), (f), (a), (b) and (e). The recall value of sitting posture (c) is 0.717. The F1-score for the classification of the sitting posture is descending in order of (g), (c), (d), (f), (a), (b), and (e). The precision of sitting posture (g) is 0.700.

3.4.2 Convolutional Neural Networks

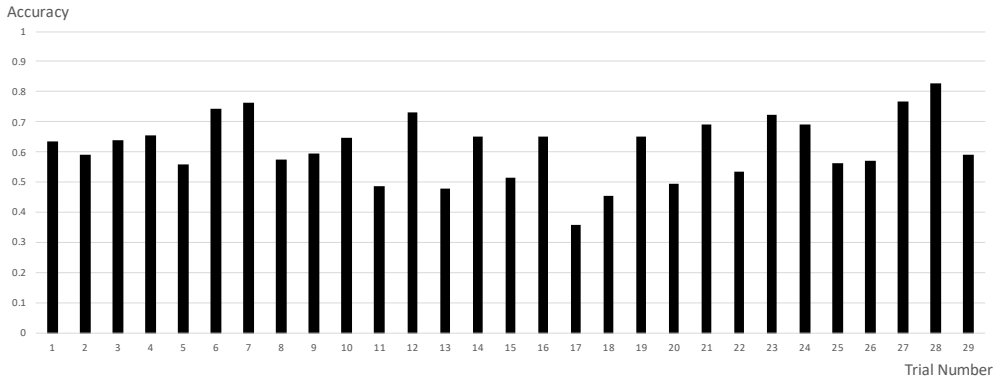


Figure 3.8 Accuracy value according to cross-validation applying Convolutional Neural Networks

The distribution of accuracy can be summarized, as shown in Figure 3.8. A total of 29 cross-validations were performed. Of those, 24 accuracy was more significant than 0.5, and 5 accuracy was less than 0.5. Accuracy was best when the 28th participant's data was used as the test data, and the rest was used as training data. When the 17th participant's data was used as the test data, and the rest was used as training data, the accuracy was worst.

Table 3.4 F1 score and accuracy of cross-validation results of convolutional neural networks

	Accuracy	F1 Score						
		a	b	c	d	e	f	g
1	0.637	0.233	0.406	0.602	0.876	0.555	0.874	0.852
2	0.590	0.297	0.480	0.875	0.411	0.478	0.819	0.783
3	0.641	0.417	0.326	0.653	0.724	0.232	0.980	0.987
4	0.656	0.581	0.406	0.813	0.688	0.491	0.937	0.744
5	0.558	0.145	0.222	0.855	0.446	0.325	0.964	0.862
6	0.745	0.158	0.568	0.890	0.829	0.673	0.996	0.858
7	0.762	0.497	0.567	0.880	0.996	0.293	0.994	0.996
8	0.575	0.429	0.417	0.531	0.500	0.330	0.898	0.821
9	0.597	0.015	0.327	0.874	0.522	0.389	0.979	0.912
10	0.647	0.112	0.392	0.669	0.883	0.501	0.998	0.779
11	0.489	0.000	0.221	0.746	0.678	0.335	0.465	0.736
12	0.734	0.656	0.347	0.913	0.756	0.483	0.910	0.921
13	0.480	0.487	0.499	0.206	0.443	0.315	0.529	0.743
14	0.653	0.560	0.345	0.924	0.602	0.380	0.703	0.990
15	0.515	0.314	0.259	0.389	0.757	0.052	0.892	0.654
16	0.651	0.286	0.617	0.795	0.792	0.357	0.974	0.901
17	0.358	0.120	0.387	0.212	0.561	0.319	0.516	0.374
18	0.455	0.542	0.199	0.469	0.650	0.029	0.405	0.664
19	0.653	0.746	0.407	0.778	0.642	0.247	0.944	0.604
20	0.496	0.125	0.448	0.517	0.252	0.120	0.994	0.828
21	0.692	0.602	0.269	0.922	0.764	0.639	0.866	0.676
22	0.537	0.570	0.480	0.404	0.639	0.261	0.556	0.842
23	0.723	0.395	0.417	0.846	0.925	0.497	0.912	0.915
24	0.692	0.499	0.428	0.754	0.891	0.281	0.992	0.989
25	0.562	0.317	0.420	0.455	0.744	0.165	0.715	0.846
26	0.571	0.423	0.382	0.703	0.585	0.251	0.794	0.905
27	0.766	0.769	0.162	0.896	0.981	0.766	0.776	0.819
28	0.826	0.688	0.634	0.826	0.987	0.870	0.862	0.989
29	0.590	0.299	0.518	0.900	0.971	0.194	0.156	0.785
Ave.	0.616	0.389	0.398	0.700	0.707	0.373	0.807	0.820
Std.	0.107	0.217	0.122	0.216	0.194	0.200	0.217	0.138
Min.	0.358	0.000	0.162	0.206	0.252	0.029	0.156	0.374
Max.	0.826	0.769	0.634	0.924	0.996	0.870	0.998	0.996

The overall results of each test are summarized with accuracy and F1 score. The results of the cross-validation of 29 pressure distribution data using modified LeNet-5 for each data are shown in Table 3.4. Accuracy is 0.358 for the minimum value and 0.826 for the maximum value. The average value is 0.616(SD=0.107). When sitting straight (a) is classified, the average F1 score is 0.389(SD=0.217). When lean forward (b) is classified, the average F1 score is 0.398(SD=0.122). When classifying Lean left (c), the average F1 score is

Table 3.5 It is a confusion matrix that sums up the results of cross-validation using Convolutional Neural Networks.

Actual posture	Predictied posture								
	a	b	c	d	e	f	g	Total	Recall
a	3027	1357	302	834	1713	201	106	7540	0.401
b	1365	3126	434	474	1351	471	319	7540	0.415
c	361	835	5200	28	521	171	424	7540	0.690
d	438	409	46	5405	731	215	282	7526	0.718
e	1209	1161	496	609	3166	567	273	7481	0.423
f	8	407	169	125	459	6250	122	7540	0.829
g	179	222	223	157	363	118	6278	7540	0.833
Total	6587	7517	6870	7632	8304	7993	7804	Accuracy	
Precision	0.460	0.416	0.757	0.708	0.381	0.782	0.804	0.616	

0.700(SD=0.216). When lean right (d) is classified, the average F1 score is 0.707(SD=0.194). When lean backward (e) is classified, the average F1 score is 0.373(SD=0.200). When sitting at the front of the chair (f), the mean F1 score is 0.807(SD=0.217). When sitting sitting-legged on the chair (g), the mean F1 score is 0.820(SD=0.138).

The overall results were summed up in the confusion matrix of Table 3.5. The accuracy value is 0.616. The recall value in (a) is 0.401, and the precision value is 0.460. The recall value in (b) is 0.415, and the precision value is 0.416. The recall value in (c) is 0.690, and the precision value is 0.757. The recall value in (d) is 0.718, and the precision value is 0.708. The recall value in (e) is 0.423, and the precision value is 0.381. The recall value in (f) is 0.829, and the precision value is 0.782. The recall value in (g) is 0.833, and the precision value is 0.804.

The mean values of the precision, recall, F1 score of each sitting posture for posture classification results are shown Figure 3.9. The sitting precision of the posture of the sitting posture is the descending order of (f), (g), (c), (d), (b), (a), and (e). The precision value of sitting posture (f) is 0.854. In the case of a recall of a sitting posture, the descending order is (g), (f), (d), (c), (e), (a)

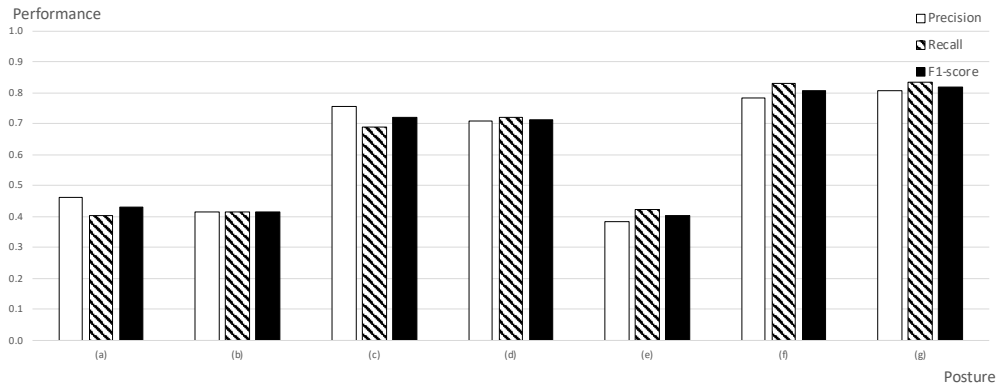


Figure 3.9 Average of precision, recall, and F1 score for each sitting posture. (a) Sitting straight, (b) Lean forward, (c) Lean left, (d) Lean right, (e) Lean backward, (f) Sitting at the front of the chair, (g) Sitting crossed-legged on the chair

and (b). The recall value of sitting posture (g) is 0.833. The F1-score for the classification of the sitting posture is descending in order of (g), (f), (d), (c), (a), (b), and (e). The precision of sitting posture (g) is 0.820.

3.4.3 Comparison of Logistic Regression Results and Convolutional Neural Networks Results

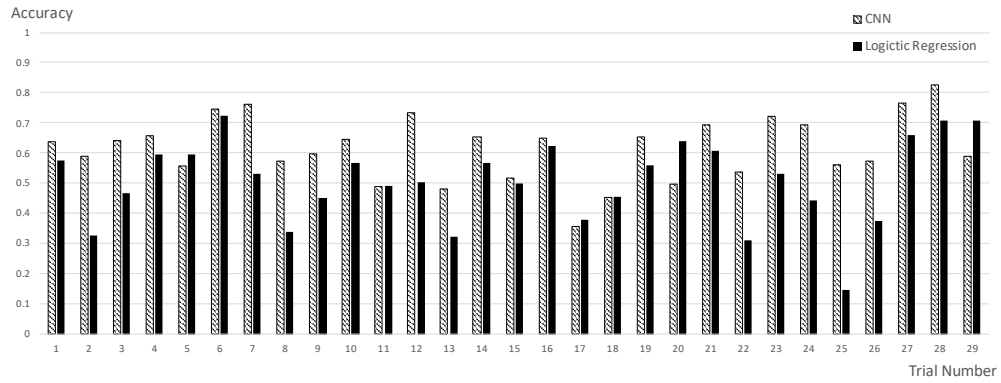


Figure 3.10 Cross-validation results comparison between logistic regression and Convolutional neural networks

Figure 3.10 shows the combination of logistic regression results and convolutional neural networks results. Except for 5, 11, 17, 20, and 29, CNN has higher accuracy than logistic regression. Paired samples t-tests was performed to confirm the statistical significance of the relationship.

Descriptive statistics of accuracy values according to each classification method can be expressed, as shown in Table 3.6. The accuracy of logistic regression is mean 0.507, median 0.531, SD 0.138, SE 0.0256. The accuracy of CNN is mean 0.616, median 0.637, SD 0.107, SE 0.0198.

Table 3.6 Descriptive statistics on accuracy values for each estimation method.

Method	N	Mean	Median	SD	SE
Logistic regression	29	0.507	0.531	0.138	0.0258
CNN	29	0.616	0.637	0.107	0.0198

According to the classification method, paired samples t-tests was conducted to see if the magnitude of accuracy was significant. Hypotheses were set to accuracy of logistic regression $<$ accuracy of CNN. The results of verifying these hypotheses by paired samples t-tests are shown in Table 3.7. The t-value was -4.75, the degree of freedom was 28, and the probability of significance was less than 0.001. The average accuracy values obtained with CNN was smaller than the average accuracy values obtained with logistic regression.

Table 3.7 Paired samples t-tests result of accuracy value

				Mean	SE	95% Confidence Interval		
Pair	**t**	**df**	**p**	**difference**	**difference**	**Lower**	**Upper**	**Cohen's d**
Logistic regression - CNN	-4.75	28.0	$<.001$	-0.108	0.0228	$-\infty$	-0.0695	-0.881

Note. H_A; Logistic regression $<$ CNN

The distribution of 95% confidence intervals and median values for the accuracy values for each classification method is shown in the plot of Figure 3.11.

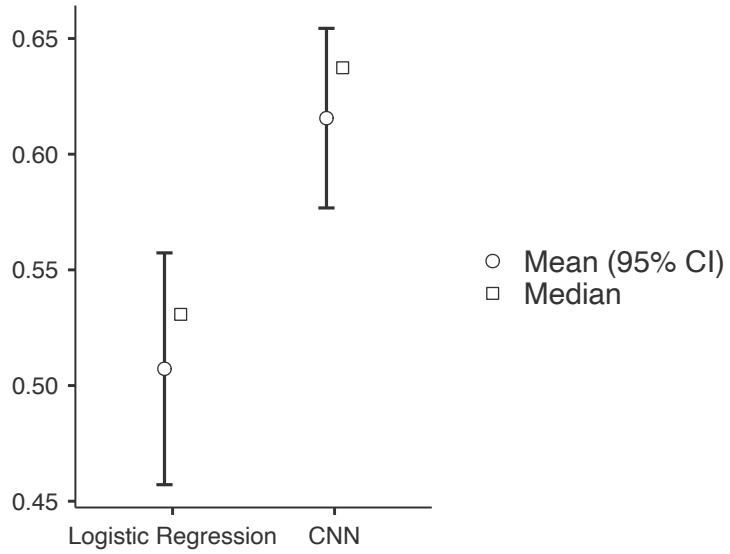


Figure 3.11 95% Confidence Interval and Median Distribution Chart for Accuracy Values

3.5 Discussion

The LeNet-5 model applied to this study is the early CNN model. Therefore, compared to the latest CNN model, the computing power required for training and testing is not so massive, which is relatively easy to use. By applying the LeNet-5 model to the classification of the pressure distribution image, it can test the application of the deep-learning method to the posture classification. The sitting posture depends on the person's body shape and habit, so the same person's body pressure distribution data cannot be used for both training and testing.

In supervised learning, the learning outcomes are used to predict outcomes for new input data. In this case, new input data cannot be used as new learning

data. In order to use the newly input data as new learning data, it is necessary to inform the new input data and the correct result, and to learn all the existing data including the new data from the beginning. Thus, 28 data were learned, and the other one was verified by cross-validation.

The performance of logistic regression and CNN model were compared based on the accuracy value. As a result, the accuracy of the CNN model was about 20% higher than the logistic regression. Moreover, the difference was statistically significant. The accuracy of the CNN model is 0.616, which is not high, but it has improved performance compared to logistic regression. It is expected to improve accuracy through the development of deep learning technology.

The range of accuracy was at least 0.358, and the maximum was 0.826. The total number of images of the body pressure distribution is 52780, which is enough to apply the deep-learning method. However, it is the limit due to the data obtained from 29 people. In terms of classification performance, the deep-learning method can be applied to the sitting posture monitoring system enough. Because it is the result of applying the early CNN model, although it is not such a high value.

As a result of 29 cross-validation, the average value of accuracy is 0.616 (SD=0.107). Because the individual difference is significant, ranging from 0.358 to 0.826. In terms of the F1 score of the individual posture classification results, (f) sitting at the front of the chair and (g) sitting-legged on the chair predicted posture with a high probability of 0.8 or higher.

(a) Sitting postures of upright, (b) lean forward, and (e) lean backward had a lower value of precision, recall, and F1-score compared to other postures. Since the lean forward and lean backward are inclined forward or backward in the sitting straight, it seems that interference between the three postures has

occurred.

In order for the classification of the posture to be good, the guidance on the posture should be proper in the process of creating training data. These postures are the positions where the center of gravity changes from the same line to the front and back direction. So, there might be a possibility that the change of the center of gravity did not appear to be significant. In addition, we aimed to take a natural posture for each individual, so we proceeded without correction for a specific posture. Therefore, it is considered that the situation of moving the center of gravity forward or backward might not have occurred. For postures that do not show high accuracy, there is a need to design experiments with more specific instructions as well as algorithm improvements.

Chapter 4

Applying Deep-learning Methods to Human Motion Analysis of Automobile Assembly Tasks

4.1 Overview

Factory workers are exposed to musculoskeletal hazards because they repeatedly work in certain positions. To prevent the occurrence of musculoskeletal disorders in workers, the risk factors for musculoskeletal disorders are measured using evaluation techniques such as OWAS, RULA, and REBA. However, there is a problem that such an evaluation requires much time and human resources. In this chapter, deep-learning is applied to estimate the workload by using the working video as input data.

4.2 Data Characteristics

4.2.1 Work-related Musculoskeletal Disorders(WMSDs) in Factory Workers

The Korean government mandated regular and occasional surveys when workers engaged in musculoskeletal burdens. Regular surveys are conducted every three years. Occasional surveys are conducted in the event of illness, the introduction of new work facilities, and changes in the working environment. The survey process is conducted by looking at the site, but also by taking a video of the work and evaluating it.

As explained in Chapter 2, OWAS is an evaluation technique developed by the Ovako Steel Company to define and evaluate workers' inappropriate working postures. It is mainly suited for the handling of heavy loads by personnel, and scores from 1 to 4 points. OWAS has the advantage of being evaluated directly in the workplace, but not suitable for repetitive tasks.

RULA is an evaluation technique developed by the University of Nottingham in the UK to prevent occupational upper extremity disease. RULA is suitable for the site where there is a lot of upper limb work, such as an automobile assembly line, and it is evaluated by a score from 1 to 7 points.

REBA is a working posture analysis tool developed to be sensitive to unpredictable work posture. REBA observes the worker's movement stage, divides body parts, and assigns points to each body part. The REBA scoring system ranges from 1 to 15 points.

4.2.2 Data Collection

The images were taken for routine and occasional investigations into the plant's musculoskeletal hazards. The results of the OWAS and RULA REBA evaluations were evaluated by ergonomic researchers on the images taken during the investigation. There are regular surveys, but also occasional surveys, so the evaluated processes may overlap.

Table 4.1 Images and evaluation results collected during the years of musculoskeletal hazard investigations on automobile factory workers

Set	Collected[a]	Filtered[b]	Video	OWAS	RULA	REBA	Remarks
2007	289	120	123	○	○		Training[c]
2009	34	32	37	○	○		Training
2012A	64	64	178	○	○		Training
2012B	81	78	165	○	○		Training
2013A	90	90	242	○	○		Training
2013B	13	11	19	○	○		Training
2013C	390	339	662	○	○	○	Training
2014	39	38	163	○	○	○	Training
2016	226	226	226	○	○	○	Test[d]

[a] the numbers of collected data

[b] survey data and video are in sync

[c] used as training data for deep-learning

[d] used as test data for deep-learning

The images and evaluation results used in this study were collected nine times from 2007 to 2016, as shown in Table 4.1. Evaluation results without videos or missing videos were excluded. In the 2007 survey, 289 data were collected, and 120 cases of evaluation data and video data were together. In the 2009 survey, 32 data were collected, and 32 cases of evaluation data and video data were together. In the 2012 A survey, 64 data were collected, and 64 cases of evaluation data and video data were together. In the 2012 B survey, 81 data were collected, and 78 cases of evaluation data and video data were together. In the 2013 A survey, 90 data were collected, and in the case of evaluation data and video data, there were 90 cases. In the 2013 B survey, 13 data were

collected, and 11 cases of evaluation data and video data together. In the 2013 C survey, 390 data were collected, and 339 cases of evaluation data and video data were together. In the 2014 survey, 39 data were collected, and 226 cases with evaluation data and video data.

From 2007 to 2013B, only OWAS and RULA data were available, and REBA was included in the 2013C, 2014, and 2016 surveys. So, 2016 data was used as test data, and the rest of the survey results were used as training data.

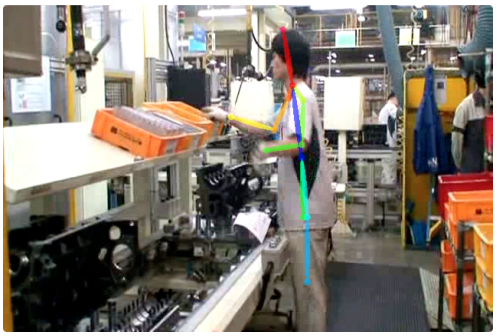
4.2.3 Data Pre-processing

The feature must be extracted from the working video to be learned and verified using the deep-learning algorithm. Since the format of the collected image is not the same, it is converted into the same form. All images were converted to AVI format with 5fps. Since only human body motion is extracted from the image with JSON, the resolution is not converted to the same shape.

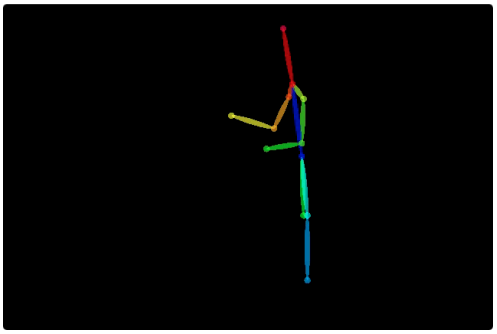
OpenPose library is used to extract human motion from videos. OpenPose is a library developed by Carnegie Mellon University that detects the human body, hands, face, and feet in a single image. OpenPose uses deep-learning technology to extract human motion from any image into a skeleton.

The process of extracting human motion from an image can be shown in Figure 4.1. Human body motion was extracted by applying OpenPose to the image converted to 5fps. We removed the background from the image and derived a JSON file.

(a)



(b)



(c)


```
{
  "version":1.0,
  "people":[
    {
      "pose_keypoints":[
        348.525,43.4867,0.82841,360.693,111.322,0.842696,394.405,137.94,0.77844
        9,0,0,0,0,0,0,341.328,123.535,0.839937,292.881,140.361,0.812262,242.146
        ,125.938,0.831102,392.042,278.309,0.741947,0,0,0,0,0,0,377.643,280.686,
        0.90131,0,0,0,0,0,0,375.248,203.289,0.964629
      ],
      "face_keypoints":[

      ],
      "hand_left_keypoints":[
        219.383,137.518,0.0850783,224.328,131.068,0.0889336,236.583,123.973,0.0
        513487,240.238,120.963,0.0899165,239.593,123.543,0.0879292,240.453,125.
        048,0.0457126,239.378,124.833,0.044294,241.528,126.553,0.1039,241.098,1
        23.758,0.064794,231.208,131.928,0.0230098,237.228,132.143,0.0706918,239
        .593,130.853,0.0537343,236.368,130.853,0.0223374,229.703,132.358,0.0137
        049,232.928,136.013,0.062506,234.648,136.013,0.062699,237.443,142.463,0
        .0235233,226.048,142.248,0.0365795,229.058,140.528,0.0458171,230.133,14
        0.743,0.0880157,224.758,140.743,0.058304
      ],
      "hand_right_keypoints":[
        0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
      ]
    }
  ]
}
```


(d)

Figure 4.1 The process of extracting human body motion from working video: (a) Working video, (b) OpenPose applied video, (c) Background removed video, (d) Only human motion saved as JSON.

4.3 Data Analysis

The learning model used Long Short-Term Memory (LSTM), a kind of Recurrent Neural Networks (RNN). The preprocessed image data was used as input data, and the evaluation results of each evaluation tool were used as output data.

In the OWAS result prediction model, 1586 data from 2007 to 2014 were used as training data, and 226 data from 2016 were used as test data. In the RULA results, 1586 data from 2007 to 2014 were used as training data, and 226 data from 2016 were used as test data. In the REBA result prediction model, 822 data from 2013 to 2014 were used as training data, and 223 data from 2016 were used as test data.

Pre-processed image data is modeled by 20 frame unit and classified by OWAS, RULA, REBA score. Since the result is a score, it can be designed as a regression model, but because there are many images with the same values, it is designed as a classification model.

The classification results were summarized in the form of a confusion matrix to derive precision, recall, F1-score, and accuracy values and evaluated for classification performance.

4.4 Results

4.4.1 OWAS Prediction Model

LSTM was used to predict the OWAS score. The results of the verification using 226 test data are expressed in the confusion matrix of Table 4.2.

Table 4.2 Confusion Matrix of OWAS Action Category(AC) Prediction

Actual	Predicted AC						
AC	1	2	3	4	Total	Recall	F1-score
1	72	59	0	0	131	0.550	0.610
2	29	48	0	0	77	0.623	0.485
3	3	10	0	0	13	0	0
4	1	4	0	0	5	0	0
Total	105	121	0	0		Accuracy	
Precision	0.686	0.397	0	0		0.531	

The accuracy of the overall classification is 0.531. The recall value of Action category 1 is 0.550, the precision value is 0.686, and the F1-score value is 0.610. The recall value of Action category 2 is 0.623, the precision value is 0.397, and the F1-score value is 0.485. The recall value of Action category 3 is 0, the precision value is 0, and the F1-score value is 0. The recall value of Action category 4 is 0, the precision value is 0, and the F1-score value is 0.

4.4.2 RULA Prediction Model

LSTM was used to predict the RULA score. The verification results using 224 test data were expressed in the confusion matrix of Table 4.3.

Table 4.3 Confusion Matrix of RULA Score Prediction

Actual	Predicted score									F1-
score	1	2	3	4	5	6	7	Total	Recall	score
1	0	0	0	0	0	0	0	0	0	0
2	0	2	0	4	0	0	6	0	0.333	0.091
3	0	20	29	34	0	4	89	2	0.325	0.369
4	0	5	13	21	0	3	42	0	0.500	0.300
5	0	3	11	16	0	3	35	2	0	0
6	0	5	8	15	0	2	32	2	0.063	0.087
7	0	3	7	8	0	2	20	0	0	0
Total	0	38	68	98	0	14	6		Accuracy	
Precision	0	0.053	0.426	0.214	0	0.143	6		0.241	

The accuracy of the overall classification is 0.241. The recall value of RULA score 1 is 0, the precision value is 0, and the F1-score value is 0. The recall

value of RULA score 2 is 0.333, the precision value is 0.053, and the F1-score value is 0.091. The recall value of RULA score 3 is 0.325, the precision value is 0.426, and the F1-score value is 0.369. The recall value of RULA score 4 is 0.500, the precision value is 0.214, and the F1-score value is 0.300. The recall value of RULA score 5 is 0, the precision value is 0, and the F1-score value is 0. The recall value of RULA score 6 is 0.063, the precision value is 0.143, and the F1-score value is 0.087. The recall value of RULA score 7 is 0, the precision value is 0, and the F1-score value is 0.

4.4.3 REBA Prediction Model

LSTM was used to predict the REBA score. The verification results using 223 test data were expressed in the confusion matrix of Table 4.4.

Table 4.4 Confusion Matrix of REBA Score Prediction

Actual	Predicted score													F1-
score	1	2	3	4	5	6	7	8	9	10	11	Total	Recall	score
1	5	3	7	1	0	0	0	0	0	0	16	0	0.313	0.189
2	3	8	9	6	0	1	3	0	0	0	30	0	0.267	0.193
3	7	7	20	5	0	1	0	0	0	0	40	0	0.500	0.313
4	8	10	15	7	0	0	3	0	0	1	45	1	0.156	0.187
5	5	8	10	2	0	0	0	0	0	0	25	0	0	0
6	2	3	7	1	0	0	1	0	0	0	14	0	0	0
7	2	3	7	2	0	1	0	0	0	1	16	0	0	0
8	1	9	7	5	0	0	0	1	0	0	23	0	0.043	0.083
9	2	2	3	0	0	0	0	0	0	0	7	0	0	0
10	2	0	2	0	0	1	0	0	0	0	5	0	0	0
11	0	0	1	1	0	0	0	0	0	0	2	0	0	0
Total	37	53	88	30	0	4	7	1	0	2	1	Accuracy		
Precision	0.135	0.151	0.227	0.233	0	0	0	1	0	0	0	0.184		

The accuracy of the overall classification is 0.184. The recall value of REBA score 1 is 0.313, the precision value is 0.135, and the F1-score value is 0.189. The recall value of REBA score 2 is 0.267, the precision value is 0.151, and the F1-score value is 0.193. The recall value of REBA score 3 is 0.500, the precision

value is 0.227, and the F1-score value is 0.313. The recall value of REBA score 4 is 0.156, the precision value is 0.233, and the F1-score value is 0.187. The recall value of REBA score 5 is 0, the precision value is 0, and the F1-score value is 0. The recall value of REBA score 6 is 0, the precision value is 0, and the F1-score value is 0. The recall value of REBA score 7 is 0, the precision value is 0, and the F1-score value is 0. The recall value of REBA score 8 is 0.043, the precision value is 1, and the F1-score value is 0.083. The recall value of REBA score 9 is 0, the precision value is 0, and the F1-score value is 0. The recall value of REBA score 10 is 0, the precision value is 0, and the F1-score value is 0. The recall value of REBA score 11 is 0, the precision value is 0, and the F1-score value is 0.

4.5 Discussion

It takes much time and human resources to evaluate OWA, RULA, REBA, which are widely used in the investigation of musculoskeletal hazards. Verification was performed to replace this time or human resources with deep-learning.

Based on the accuracy, the accuracy of the OWAS score prediction was 0.531, the accuracy of the RULA score prediction was 0.241, the accuracy of the REBA score prediction was 0.184, and the performance of the OWAS score prediction was better than that of the RULA and REBA.

The performance of OWAS, RULA, and REBA scores is not bad if each classification performance is based on a baseline. The accuracy of the model for predicting OWAS scores from working images is 0.531. Since the model predicts 1 to 4 action category, the baseline can be seen as 0.250. Since the accuracy value is larger than the baseline, it is worth introducing deep-learning to the OWAS evaluation. The accuracy of the model for predicting RULA scores from

Table 4.5 OWAS, RULA, REBA score distribution of data used for training and test

	OWAS		RULA		REBA	
	Training	Test	Training	Test	Training	Test
1	788	131	2	0	47	16
2	725	77	140	6	259	30
3	32	13	553	89	356	40
4	44	5	614	42	116	45
5	-	-	110	35	2	25
6	-	-	107	32	13	14
7	-	-	63	20	9	16
8	-	-	-	-	10	23
9	-	-	-	-	0	7
10	-	-	-	-	9	5
11	-	-	-	-	4	2
12	-	-	-	-	0	0
13	-	-	-	-	0	0

working images is 0.241. Since the model predicts 1 to 7, the baseline can be seen as 0.143. Since the accuracy value is larger than the baseline, it may be worth introducing deep-learning to the RULA evaluation. The accuracy of the model for predicting REBA scores from working images is 0.194. Since the model predicts 1 to 15, the baseline can be seen as 0.077. Since the accuracy value is larger than the baseline, it is worth introducing deep-learning into the REBA evaluation.

The performance of OWAS, RULA, and REBA through deep-learning is better than the baseline, but 0.531, 0.241, and 0.194 are not very good because the accuracy of general classification models is often more than 0.9. The reason for the poor performance is the characteristics of the data used for training. Table 4.5 summarizes the distribution of data used for OWSA, RULA, and REBA predictions. The training data used for OWAS prediction was 788 data with one score and 725 data with score 2, which accounted for 95.2% of the total training data. Similarly, in the case of RULA, the data with score three and the data with score 4 occupy 73.4% of the total training data. There were only two data points with a score 1. In the case of REBA, data with a score 2

Table 4.6 Ratio of data set and classification result of the top two items of OWAS, RULA, and REBA

	Training	Test	Predicted
OWAS Top 2	0.952	0.920	1
RULA Top 2	0.734	0.585	0.741
REBA Top 2	0.745	0.314	0.632

and a score 3 account for 74.5% of the total training data. In the case of score 9, score 12, score 13, there is no training data. Since there is a bias in the learned data, it can be said that it also influenced the result of verification using the test data.

Table 4.6 shows the distribution of the top two cases among the classification results. OWAS was classified 100% into score 1 and score 2, which accounted for 95.2% of the training data. In RULA, 74.1% of the test data was classified into score 3 and score 4, which accounted for 73.4% of the training data. In test data used for RULA analysis, the ratio of score 3 to score 4 was 58.4%. In the case of REBA, scores 2 and 3 accounted for 74.5% of the total training data, but they accounted for 63.3%. In test data used for REBA analysis, score two and score 3 were 31.1% of the total test data. As a result of classifying the test data, the items which occupy the majority of the training data were found to be classified more than the ratio of the actual values.

Deep-learning requires many data. However, based on the results of applying deep-learning to OWAS, RULA, and REBA, the amount is not necessary. If the training data is not enough for each case, the classification for the case is not functional. On the contrary, if the training data for a particular case is excessively large, the ratio classified to the case increases even though the case is not. Each case requires a lot of different data. If there is enough data to represent each case, the performance of deep-learning is maximized. As a result

of applying deep-learning to OWAS, RULA, and REBA evaluation, it can be seen that the result of classifying the OWAS score that has more than 30 data per case and the case where there are few cases to classify is the best even though there is bias among the cases.

Chapter 5

Estimation of Hand Anthropometric Dimensions Using a Deep-learning Method

5.1 Overview

Body size information is one of the necessary information for making various products. Such body size information can play an essential role in crime resolution. Information such as the suspect's stature may reduce the scope of the suspect. There is also a study using linear regression to estimate the stature from the size of a portion of the hand(Jee and Yun, 2015). In this chapter, RNN and RGLM corresponding to the regression of deep-learning methods are applied to estimate the stature from the size of the part of the hand and compared with linear regression.

5.2 Data Characteristics

5.2.1 Size Korea; A National Anthropometric Survey of Korea

Humans use many things, equipment, tools, and facilities in everyday life or work. If these objects do not fit the body of the person who uses them, they are not convenient and productive, but they cause accidents. Therefore, product size fitted for body size is very important not only for productivity, but also for safety. In order to design the tool or work environment that we use, the human body dimension data is the most basic.

Anthropometry is defined as measuring the physical properties of a human body, including various dimensions of the body, volume, center of gravity, inertia, and mass of each region(Herron, 2000). Since 1979, the Korean government has carried out seven projects for measurements of human body size by 2015 with a period of five to seven years. In the 5th size Korea project, measurement data were extended by dynamic characteristics, head, and foot shape measurement using a 3D scanner.

From the first human-size measurement project conducted in 1979, hand-related dimensions have been measured. According to Table 5.1, only 5-18 dimensions of hands were measured in these anthropometric measurements. However, it is difficult to express complex shapes of hands with these dimensions alone. Besides, when designing hand related items, there was a lack of data on hand related dimensions, so data of foreign institutions, especially NASA data, were used to be adjusted based on Korean height.

Under the necessity of measuring the hand-related static dimensions, the size measurement of the hand-related body in 2008 was carried out by Size Korea. The hand-related static dimensions were measured according to the 58 mea-

Table 5.1 Hand dimensions measured in Size Korea project

Round	Year	Method	Count	Dimensions of hand
1	1979	Manual	10	Hand length, Palm length, Index finger length, Middle finger length, Ring finger length, Little finger length, Thumb finger length, Hand breadth, Maximum hand breadth, Hand thickness
2	1986	Manual	8	Hand length, Palm length, Hand breadth at metacarpals, Hand breadth across thumb, Index finger length, Hand thickness, GRIP I, GRIP II
3	1992	Manual	5	Hand length, Hand breadth, Hand circumference, Palm length perpendicular, Hand thickness
4	1997	Manual	9	Hand length, Hand breadth, Hand circumference, Palm length Perpendicular, Hand thickness, Wrist circumference, Maximum hand circumference, Middle finger length, Maximum hand breadth
5	2003 - 2004	Manual	9	Hand length, Palm length perpendicular, Hand breadth at metacarpals, Index finger length, Index finger breadth-proximal, Index finger breadth-distal, Hand thickness, Inner grip circumference, Hand circumference
6	2010 - 2013	manual	9	Hand Length, Palm length perpendicular, Hand breadth, Index finger length, Index finger breadth-proximal, Index finger breadth-distal, Hand thickness, Inner grip circumference, Hand circumference
		3D	18	Hand length, Index finger length, Middle finger length, Ring finger length, Little finger length, Thumb-wrist length, Thumb finger length, Thumb finger-index finger length, Hand circumference, Thumb finger circumference-upper, Thumb finger circumference-lower, Index finger breadth-lower, Middle finger breadth-lower, Ring finger breadth-lower, Little finger breadth-lower, Index finger breadth-upper, Middle finger breadth-upper, Little finger breadth-upper
7	2015	Manual	9	Hand length, Palm length perpendicular, Hand breadth with thumb, Index finger length, Index finger breadth-proximal, Index finger breadth-distal, Hand thickness, Inner grip circumference, Hand circumference

surement items and the measurement protocol presented in Choi et al. (2006)'s 'Development of Hand Part Measurement Protocol for Glove Design' report. A total of 63 items were measured by adding height, weight, upper arm circumference, lower arm circumference, and arm length to 58 measurement items in order to examine the relationship between these hand-related dimensions and typical body part measurement items.

5.2.2 Hand Anthropometric Measurement Data

In the hand-related measurement performed in 2008, the sample was extracted by sex and age. The data for 63 items in Table 5.2 were collected from 840 males and females from ages 4 to 83. This data is available in Size Korea homepage

(http://sizekorea.kr). The entire data is in an Excel file, and all lengths are measured in mm.

Table 5.2 List of parts for human body measurement

Group	Count	Dimensions
hand	58	Hand length, Palm length perpendicular, Thumb length, Index finger length, Middle finger length, Ring finger length, Little finger length, First phalanx length-thumb, First phalanx length-index finger, First phalanx length-middle finger, First phalanx length-ring finger, First phalanx length-little finger, Second phalanx length-thumb, Second phalanx length-index finger, Second phalanx length-middle finger, Second phalanx length-ring finger, Second phalanx length-little finger, Thumb-index finger first crease line length, Thumb-middle finger first crease line length, Thumb-ring finger first crease line length, Thumb-little finger first crease line length, Capitate-thumb first crease line length, Capitate-index finger first crease line length, Capitate-middle finger first crease line length, Capitate-ring finger first crease line length, Capitate-little finger first crease line length, Radial styloid-thumb fingertip length, Hand breadth, Maximum hand breadth, Wrist breadth, Maximum finger span breadth, Thumb breadth-proximal, Index finger breadth-proximal, Middle finger breadth-proximal, Ring finger breadth-proximal, Little finger bradth-proximal, Index finger breadth-distal, Middle finger breadth-distal, Ring finger breadth-distal, Little finger breadth-distal, Middle finger phalanx length-dorsal, Middle finger phalanx length-dorsal, Middle finger phalanx length-dorsal, Middle finger phalanx length-dorsal-flexed, Middle finger phalanx length-dorsal-flexed, Middle finger phalanx length-dorsal-flexed, Maximum hand thickness, Hand thickness, Inner grip circumference, Thumb circumference-proximal, Index finger circumference-proximal, Middle finger circumference-proximal, Ring finger circumference-proximal, Little finger circumference-proximal, Middle finger circumference-proximal-flexed, Wrist circumference, Hand circumference, Maximum hand circumference
Body	5	Stature, Weight, Upper arm circumference, Lower arm circumference, Arm length

5.2.3 Data Selection and Hand Dimension

The purpose of this study is to investigate the relationship between hand related measurements and stature. Therefore, the analysis was conducted on the data of adults who have already grown. A total of 327 data were selected for the whole of males and females. Among them, there are 173 men from 20 to 75 years old and 154 women from 20 to 83 years old.

The measurements included a total of 29 variables, indicated in Table 5.3 and Figure 5.1, they are lengths, breadths, thicknesses, and circumferences of fingers, phalanges, palms, and wrists(Aboul-Hagag et al., 2011; Ishak et al., 2012; Jee and Yun, 2015; Kanchan and Krishan, 2011).

Table 5.3 List of 29 hand regions selected to estimate stature in Jee and Yun (2015)

Type	Hand diimension	Abbreviation	Definition
Length	Hand length	HL	The distance from the middle of inter stylion to the tip of middle finger
	Palm length	PL	The distance from the middle of inter stylion to the proximal flexion crease of the middle finger
	Thumb; index; middle; ring; little finger length	1DL, 2DL, 3DL, 4DL, 5DL	The distance from the proximal flexion crease of the finger to the tip of the respected finger
	Thumb; index; middle; ring; little finger proximal phalange length	1DT2L, 2DI3L, 3DM3L, 4DR3L, 5DL3L	The distance from the proximal interphalangeal joint crease to metacarpophalangeal joint crease of each finger
	Thumb; index; middle; ring; little finger middle phalange length	2DI2L, 3DM2L, 4DR2L, 5DL2L	The distance from the distal interphalangeal joint crease to the proximal interphalangeal joint crease
	Index; middle; ring; little finger distal phalange length	1DT1L, 2DI1L, 3DM1L, 4DR1L, 5DL1L	The distance from the most forwarding projecting point on the tip of each finger to distal interphalangeal joint crease of each finger
Breadth	Hand breadth	HB	The distance from the most lateral point on the head of the 2D metacarpal to the most medial point on the head of 5D metacarpal
	Maximum hand breadth	MHB	The distance from the most lateral point on the head of the 1D metacarpal to the most medial point on the head of 5D metacarpal with closing fingers
	Wrist breadth	WB	The distance from the most lateral point on the wrist to the most medial point of wrist
Circumference	Hand circumference	HC	The superficial distance around the edge of metacarpal
	Maximum hand circumference	MHC	The maximum superficial distance around the edge of the hand with closing fingers
	Wrist circumference	WC	The superficial distance around the edge of the wrist
Thickness	Hand thickness	HT	The distance from the back of the middle finger to the most medial point of palm
	Maximum hand thickness	MHT	The maximum distance from the back of the hand to the most projected point of abductor pollicis brevis

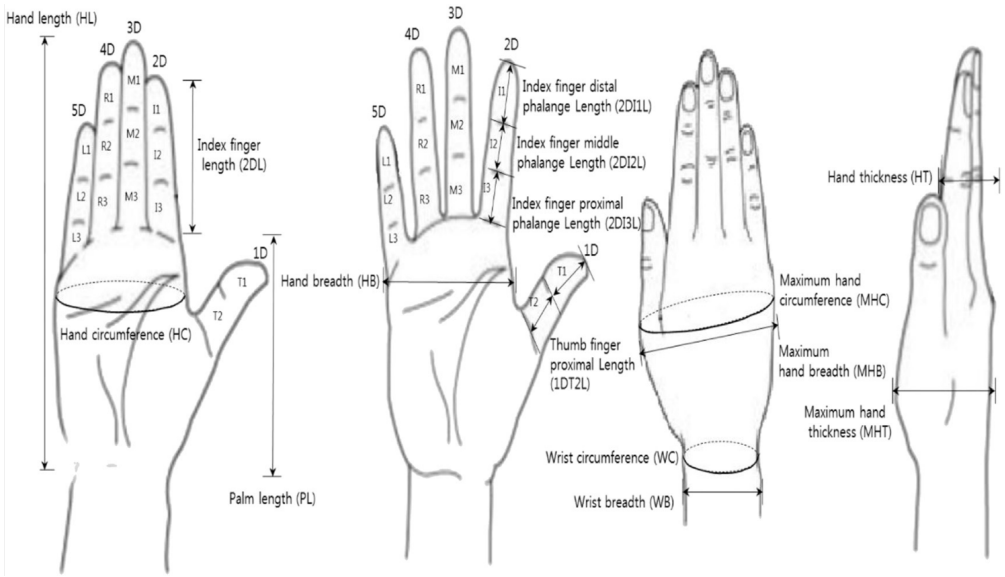


Figure 5.1 Various hand landmarks for measurement (Jee and Yun, 2015)

In Jee and Yun (2015), a variable combination was derived by stepwise regression analysis to estimate the stature using the size of each part of the hand. The hand size combinations for estimating the stature of both men and women were WC, PL, 3DL, and 3D3L, with an R2 value of 0.642 with an estimated error of 5.719 cm. Hand size combinations for estimating male stature were HL, 3DM2L, and PL, with an R2 value of 0.425 with an estimated error of 4.819 cm. Hand size combinations for estimating female stature were HL, MHB, 3DM1L, and 1DT1L, with an R2 value of 0.418 with an estimated error of 5.080 cm. In this study, regression and deep-learning were applied to three hand combinations derived from Jee and Yun (2015).

5.2.4 Training Data and Test Data

Thirty-three data corresponding to 10% of the total 327 data were randomly selected and used as test data. The remaining 294 data were used as training

data. One data has 29 hand dimension and stature value for one person.

5.3 Data Analysis

In this study, linear regression, RNN, and RGLM were used to estimate stature in the length of various parts of the hand. Models were estimated to estimate stature from 29 parts of the hand, WC, PC, 3DL, 3DM3L combination, HL, 3DM2L, PL combination, HL, MHB, 3DM1L, 1DT1L combination, and all combinations. Since there are three estimation methods used, 99 models were created to estimate the stature. The performance of their accuracy is compared based on Relative Absolute Error(RAE), Relative Squared Error(RSE), Mean Absolute Percentage Error(MAPE), Mean Absolute Scaled Error(MASE), Root Mean Square Error(RMSE), Mean Absolute Error(MAE), and Mean Squared Error(MSE) in Table 5.4.

Table 5.4 Regression performance evaluation metrics

Metric	Full name	Formula
RAE	Relative Absolute Error	$\mathrm{RAE} = \frac{\sum_{i=1}^{n} \lvert p_i - a_i \rvert}{\sum_{i=1}^{n} \lvert \bar{a} - a_i \rvert}$
RSE	Relative Squared Error	$\mathrm{RSE} = \frac{\sum_{i=1}^{n} (p_i - a_i)^2}{\sum_{i=1}^{n} (\bar{a} - a_i)^2}$
MAPE	Mean Absolute Percentage Error	$\mathrm{MAPE} = \frac{100}{n} \sum_{t=1}^{n} \left\lvert \frac{A_t - F_f}{A_t} \right\rvert$
MASE	Mean Absolute Scaled Error	$\mathrm{MASE} = \frac{\frac{1}{J} \sum_{j} \left\lvert e_j \right\rvert}{\frac{1}{T-1} \sum_{t=2}^{T} \left\lvert Y_t - Y_{t-1} \right\rvert}$
RMSE	Root Mean Square Error	$\mathrm{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2}$
MAE	Mean Absolute Error	$\mathrm{MAE} = \frac{1}{n} \sum_{j=1}^{n} \left\lvert y_j - \hat{y_j} \right\rvert$
MSE	Mean Squared Error	$\mathrm{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$

RAE and RSE are similar in that they use the difference between actual and predicted values. However, RAE uses absolute values, and RSE uses square values. MAPE is the average ratio of errors to actual values. Since RAE, RSE, and MAPE are metric expressed as ratios, performance can be compared even

when units are different. MASE measures how much difference there is from the usual variation, which is useful for predicting high and low volatility indicators. Since MAE gives less weight to outliers, it can be used when there are many outliers. MSE is often used as a loss function. RMSE is the square root of MSE, and MSE and RMSE are relatively heavy weighting factors for large errors. MASE, MAE, and RMSE use the same units as the measured value, so they cannot be applied when the object is different. Since MSE is a squared value, it can only be used for the same object, such as MASE, MAE, and RMSE.

In this study, three algorithms are used to estimate the stature in 33 datasets, each of 29 areas of the hand, three combinations derived from Jee and Yun (2015), and the entire area of the hand. It is necessary to compare the estimation performance of each dataset for the same algorithm and the estimation performance of the algorithm for a specific dataset. Rate-based evaluation metrics, RAE, RSE, and MAPE, were used to compare the estimated performance of each dataset for the same algorithm. MASE, MAE, RMSE, and MSE, which are evaluation metrics based on existing units, were used to compare the estimated performance according to algorithms for specific datasets.

5.4 Result

Linear regression, RNN, and RGLM were applied to estimate stature with 33 data sets. Thus, a total of 99 stature prediction models were created. RAE, RSE, MAPE, MASE, RMSE, MAE, and MSE were applied as evaluation metrics for measuring the accuracy of the stature prediction model.

Table 5.5 compares the estimation performance between data sets according to the estimation method with RAE, RSE, MAPE, MASE, RMSE, MAE, and MSE values.

Table 5.5 RAE, RSE, MAPE, MASE, RMSE, MAE, MSE value of CNN, RNN, RGLM

	RAE			RSE			MAPE			MASE			RMSE			MAE			MSE		
	LR	RNN	RGLM	LR	RNN	RGLM	LR	RNN	RGLM	LR	RNN	RGLM	LR	RNN	RGLM	LR	RNN	RGLM	LR	RNN	RGLM
WC	0.915	0.688	0.665	0.901	0.599	0.522	5.261	3.954	3.822	0.607	0.457	0.441	106	86	80	86	65	62	11131	7398	6446
HC	1.029	1.079	0.825	1.149	1.326	0.705	4.840	5.073	3.881	0.856	0.897	0.686	97	104	76	79	83	64	9420	10870	5778
HB	0.821	0.833	0.823	0.696	0.699	0.758	3.822	3.876	3.829	0.579	0.587	0.580	75	75	78	62	63	63	5625	5652	6131
MHB	1.473	0.791	0.970	7.094	0.756	1.141	7.968	4.276	5.249	0.884	0.475	0.583	290	95	116	131	70	86	84209	8974	13548
WB	1.262	1.079	0.735	1.864	1.453	0.735	5.405	4.621	3.149	0.809	0.692	0.471	108	96	68	88	76	52	11737	9152	4630
HT	1.521	1.116	0.926	2.588	1.237	0.883	6.692	4.910	4.075	1.243	0.912	0.757	145	100	85	110	80	67	21040	10059	7179
MHT	2.948	1.058	1.100	22.085	1.138	2.233	12.696	4.556	4.739	2.060	0.739	0.769	414	94	131	206	74	77	170986	8812	17292
HL	1.366	0.875	1.013	10.692	0.700	2.440	6.408	4.106	4.753	0.877	0.562	0.651	317	81	152	105	68	78	100609	6589	22963
MHC	1.101	0.805	0.839	1.205	0.743	0.777	6.108	4.465	4.654	0.781	0.571	0.595	119	93	95	99	73	76	14126	8715	9116
PL	0.685	0.681	0.815	0.478	0.515	0.612	3.193	3.173	3.798	0.506	0.503	0.602	66	69	75	53	52	63	4393	4733	5626
1DL	1.130	1.076	0.836	1.518	1.188	0.670	5.047	4.806	3.734	0.829	0.789	0.613	105	93	70	83	79	61	11022	8629	4865
2DL	0.794	0.764	0.691	0.734	0.602	0.501	3.868	3.718	3.362	0.561	0.539	0.487	82	74	67	63	61	55	6662	5462	4544
3DL	0.848	0.950	0.721	0.706	0.833	0.546	3.781	4.235	3.216	0.629	0.705	0.535	75	81	66	61	68	52	5592	6603	4324
4DL	1.062	0.977	0.738	1.332	1.211	0.673	4.680	4.304	3.255	1.029	0.946	0.715	94	89	66	76	70	53	8756	7958	4421
5DL	1.357	0.750	0.798	1.952	0.596	0.584	5.669	3.136	3.333	0.985	0.545	0.579	115	63	63	92	51	54	13168	4021	3938
1D2L	1.708	1.117	0.869	3.838	1.359	0.729	8.126	5.312	4.136	1.311	0.857	0.667	172	102	75	131	85	66	29412	10418	5590
2D3L	1.611	1.125	0.939	2.719	1.343	0.866	8.254	5.763	4.810	1.026	0.716	0.598	161	113	91	134	94	78	25977	12834	8276
3D3L	1.376	1.180	0.945	2.362	1.659	0.902	6.361	5.456	4.369	1.071	0.919	0.735	137	115	85	104	89	72	18822	13219	7186
4D3L	1.558	1.292	0.999	2.823	1.932	1.074	7.573	6.283	4.857	1.095	0.908	0.702	155	128	96	124	103	79	24011	16430	9136
5D3L	1.982	1.374	0.944	4.938	2.281	0.966	8.176	5.670	3.894	1.588	1.101	0.756	177	120	78	135	93	64	31213	14421	6106
1D1L	1.102	1.019	0.746	1.428	1.506	0.629	5.478	5.066	3.709	0.768	0.711	0.520	110	113	73	88	82	60	12043	12705	5307
2D2L	1.149	1.006	0.779	1.494	1.194	0.603	6.304	5.518	4.273	0.912	0.798	0.618	124	111	79	102	89	69	15424	12325	6225
3D2L	1.165	1.047	0.953	1.734	1.414	0.937	6.077	5.461	4.971	0.929	0.834	0.760	127	115	93	98	88	80	16103	13136	8702
4D2L	1.523	0.908	0.907	3.883	1.155	0.982	7.707	4.596	4.592	1.163	0.694	0.693	189	103	95	126	75	75	35702	10623	9025
5D2L	3.000	1.404	0.960	9.218	1.873	1.013	16.023	7.500	5.129	2.580	1.207	0.826	307	138	102	260	122	83	94228	19151	10357
2D1L	1.133	1.127	0.997	1.530	1.238	0.930	5.761	5.727	5.069	0.888	0.883	0.781	123	111	96	94	94	83	15245	12338	9262
3D1L	1.511	0.883	0.748	2.847	1.010	0.750	7.305	4.269	3.618	1.062	0.621	0.526	150	89	77	121	70	60	22397	7947	5900
4D1L	1.122	0.769	0.931	1.422	0.711	0.869	5.872	4.024	4.869	0.752	0.515	0.624	116	82	91	96	66	80	13560	6782	8287
5D1L	1.485	0.998	0.956	2.758	1.146	0.931	6.339	4.260	4.082	1.186	0.797	0.764	144	93	84	103	69	67	20829	8653	7036
Set1[a]	0.960	0.879	0.882	2.983	0.852	2.565	3.933	3.604	3.615	0.689	0.631	0.633	144	77	134	65	60	60	20826	5947	17910
Set2[b]	0.988	1.247	0.807	1.056	1.817	0.686	3.178	4.011	2.595	0.665	0.839	0.543	65	85	52	53	67	43	4235	7286	2749
Set3[c]	1.565	1.397	1.047	2.705	2.052	1.205	3.213	2.867	2.149	1.165	1.040	0.780	64	56	43	51	45	34	4118	3124	1835
Set4[d]	0.593	1.088	0.569	0.413	1.364	0.373	2.949	5.415	2.833	0.394	0.723	0.378	62	112	59	49	90	47	3806	12575	3442

[a]WC, PL, 3DL, 3D3L, [b]HL, 3D2L, PL, [c]HL, MHB, 3D1L,[d]whole hand parts 1D1L

5.4.1 Comparison of Relative Absolute Error(RAE)

Descriptive statistics of RAE values according to each estimation method can be expressed, as shown in Table 5.6. The RAE of linear regression is mean 1.329, median 1.165, SD 0.528, SE 0.092. The RAE of RNN is mean 1.012, median 1.019, SD 0.199, SE 0.0346. The RAE of RGLM is mean 0.863, median 0.869, SD 0.121, SE 0.0211.

Table 5.6 Descriptive statistics on RAE values for each estimation method.

Method	N	Mean	Median	SD	SE
Linear Regression	33	1.329	1.165	0.528	0.092
RNN	33	1.012	1.019	0.199	0.0346
RGLM	33	0.863	0.869	0.121	0.0211

According to the estimation method, paired samples t-tests was conducted to see if the magnitude of RAE was significant. The results of paired samples t-tests are shown in Table 5.7.

Table 5.7 Paired samples t-tests result of RAE value

				Mean	SE	95% Confidence Interval		
Pair	t	df	p	difference	difference	Lower	Upper	Cohen's d
Linear Regression - RNN	3.94	32	$<.001$	0.317	0.0804	0.1808	∞	0.686
Linear Regression - RGLM	5.81	32	$<.001$	0.466	0.0802	0.3299	∞	1.011
RNN - RGLM	4.53	32	$<.001$	0.149	0.0328	0.0931	∞	0.789

Note. H_A; Measure 1 >Measure 2

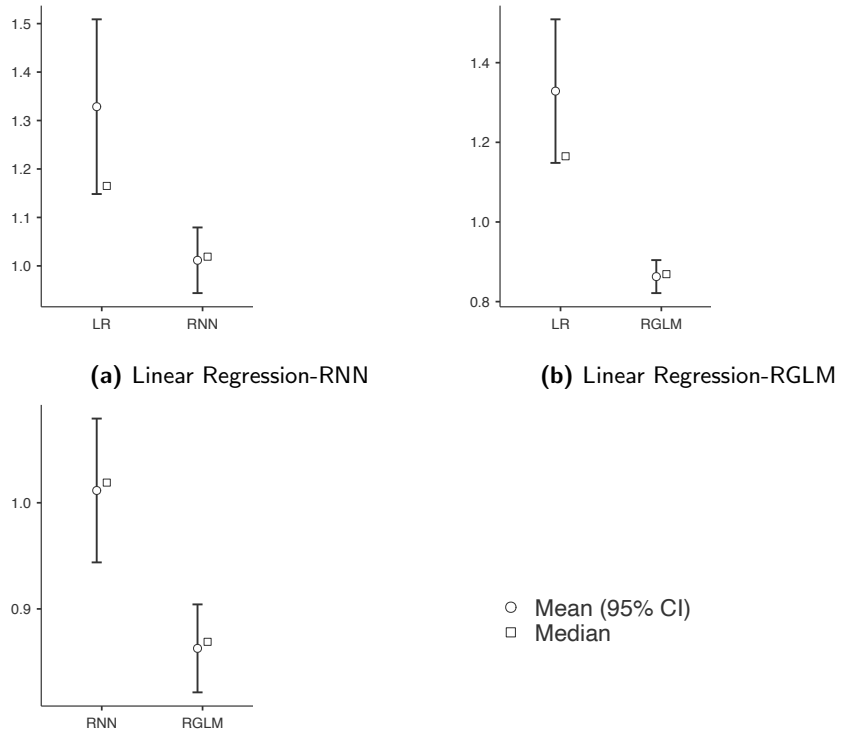
A paired samples t-tests was performed to compare whether the RAE value for the linear regression result was higher than the RAE value for the RNN result. Since the t value is 3.94, the RAE value for the linear regression result is higher than the RAE value for the RNN result under the statistical significance level.

A paired samples t-tests was performed to compare whether the RAE value for the linear regression result was higher than the RAE value for the RGLM

result. Since the t value is 5.81, the RAE value for the linear regression result is higher than the RAE value for the RGLM result under the statistical significance level.

A paired t-test was performed to compare whether the RAE value for the RNN result was higher than the RAE value for the RGLM result. Since the t value is 4.53, the RAE value for the RNN result is higher than the RAE value for the RGLM result under the statistical significance level.

The distribution of 95% confidence intervals and median values for the RAE values for each estimation method is shown in the plot of Figure 5.2.



(a) Linear Regression-RNN

(b) Linear Regression-RGLM

(c) RNN-RGLM

Figure 5.2 95% Confidence Interval and Median Distribution Chart for RAE Values

5.4.2 Comparison of Relative Squared Error(RSE)

Descriptive statistics of RSE values according to each estimation method can be expressed, as shown in Table 5.8. The RSE of linear regression is mean 3.186, median 1.864, SD 4.13, SE 0.7189. The RSE of RNN is mean 1.197, median 1.194, SD 0.46, SE 0.0801. The RSE of RGLM is mean 0.933, median 0.777, SD 0.514, SE 0.0894.

Table 5.8 Descriptive statistics on RSE values for each estimation method.

Method	N	Mean	Median	SD	SE
Linear Regression	33	3.186	1.864	4.13	0.7189
RNN	33	1.197	1.194	0.46	0.0801
RGLM	33	0.933	0.777	0.514	0.0894

According to the estimation method, paired samples t-tests was conducted to see if the magnitude of RSE was significant. The results of paired samples t-tests are shown in Table 5.9.

Table 5.9 Paired samples t-tests result of RSE value

				Mean	SE	95% Confidence Interval		
Pair	t	df	p	difference	difference	Lower	Upper	Cohen's d
Linear Regression - RNN	2.77	32	0.005	1.989	0.717	0.7739	∞	0.483
Linear Regression - RGLM	3.41	32	$<.001$	2.253	0.661	1.1343	∞	0.594
RNN - RGLM	2.17	32	0.019	0.264	0.122	0.0575	∞	0.377

Note. H_A; Measure 1 >Measure 2

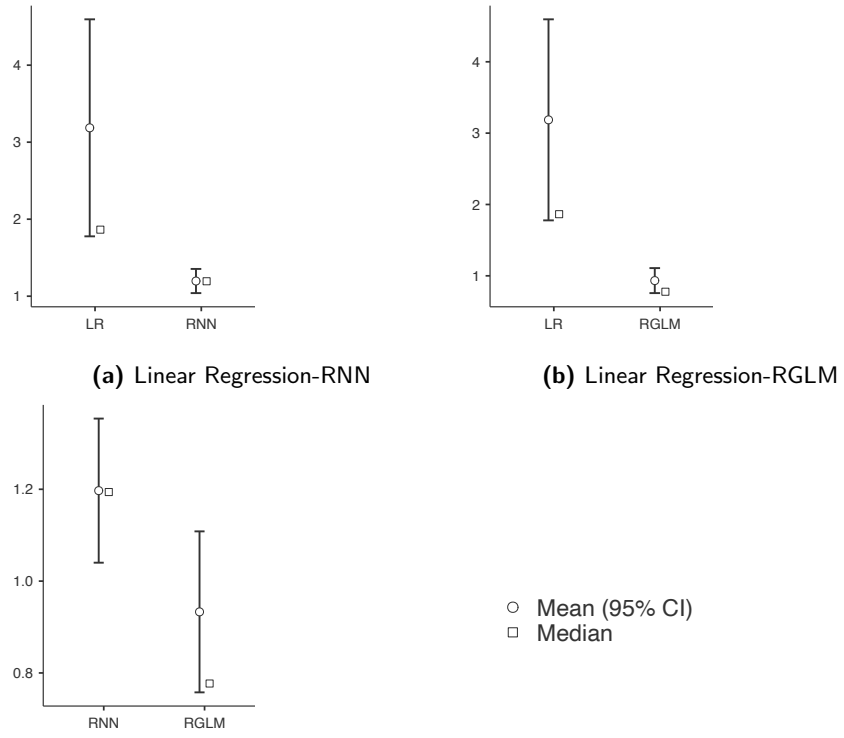
A paired samples t-tests was performed to compare whether the RSE value for the linear regression result was higher than the RSE value for the RNN result. Since the t value is 2.77, the RSE value for the linear regression result is higher than the RSE value for the RNN result under the statistical significance level.

A paired samples t-tests was performed to compare whether the RSE value for the linear regression result was higher than the RSE value for the RGLM

result. Since the t value is 3.41, the RSE value for the linear regression result is higher than the RSE value for the RGLM result under the statistical significance level.

A paired samples t-tests was performed to compare whether the RSE value for the RNN result was higher than the RSE value for the RGLM result. Since the t value is 3.41, the RSE value for the RNN result is higher than the RSE value for the RGLM result under the statistical significance level.

The distribution of 95% confidence intervals and median values for the RSE values for each estimation method is shown in the plot of Figure 5.3.



(a) Linear Regression-RNN

(b) Linear Regression-RGLM

(c) RNN-RGLM

Figure 5.3 95% Confidence Interval and Median Distribution Chart for RSE Values

5.4.3 Comparison of Mean Absolute Percentage Error(MAPE)

Descriptive statistics of MAPE values according to each estimation method can be expressed, as shown in Table 5.10. The MAPE of linear regression is mean 6.18, median 5.87, SD 2.665, SE 0.464. The MAPE of RNN is mean 4.67, median 4.56, SD 0.973, SE 0.169. The MAPE of RGLM is mean 4.01, median 3.89, SD 0.776, SE 0.135.

Table 5.10 Descriptive statistics on MAPE values for each estimation method.

Method	N	Mean	Median	SD	SE
Linear Regression	33	6.18	5.87	2.665	0.464
RNN	33	4.67	4.56	0.973	0.169
RGLM	33	4.01	3.89	0.776	0.135

According to the estimation method, paired samples t-tests was conducted to see if the magnitude of MAPE was significant. The results of paired samples t-tests are shown in Table 5.11.

Table 5.11 Paired samples t-tests result of MAPE value

				Mean	SE	95% Confidence Interval		
Pair	t	df	p	difference	difference	Lower	Upper	Cohen's d
Linear Regression - RNN	3.96	32	<.001	1.517	0.383	0.869	∞	0.69
Linear Regression - RGLM	5.56	32	<.001	2.171	0.391	1.509	∞	0.967
RNN - RGLM	4.33	32	<.001	0.654	0.151	0.398	∞	0.753

Note. H_A; Measure 1 >Measure 2

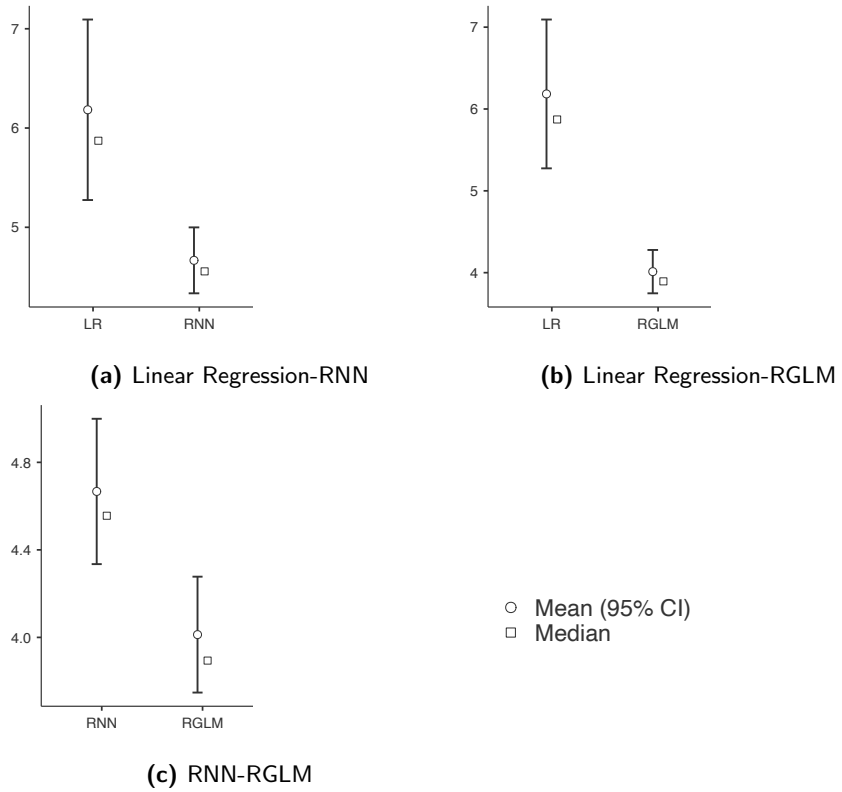
A paired samples t-tests was performed to compare whether the MAPE value for the linear regression result was higher than the MAPE value for the RNN result. Since the t value is 3.96, the MAPE value for the linear regression result is higher than the MAPE value for the RNN result under the statistical significance level.

A paired samples t-tests was performed to compare whether the MAPE value for the linear regression result was higher than the MAPE value for the

RGLM result. Since the t value is 5.56, the MAPE value for the linear regression result is higher than the MAPE value for the RGLM result under the statistical significance level.

A paired samples t-tests was performed to compare whether the MAPE value for the RNN result was higher than the MAPE value for the RGLM result. Since the t value is 4.33, the MAPE value for the RNN result is higher than the MAPE value for the RGLM result under the statistical significance level.

The distribution of 95% confidence intervals and median values for the MAPE values for each estimation method is shown in the plot of Figure 5.4.



(a) Linear Regression-RNN

(b) Linear Regression-RGLM

(c) RNN-RGLM

Figure 5.4 95% Confidence Interval and Median Distribution Chart for MAPE Values

5.4.4 Comparison of Mean Absolute Scaled Error(MASE)

Descriptive statistics of MASE values according to each estimation method can be expressed, as shown in Table 5.12. The MASE of linear regression is mean 0.984, median 0.888, SD 0.433, SE 0.0754. The MASE of RNN is mean 0.749, median 0.723, SD 0.186, SE 0.0324. The MASE of RGLM is mean 0.635, median 0.624, SD 0.111, SE 0.0194.

Table 5.12 Descriptive statistics on MASE values for each estimation method.

Method	N	Mean	Median	SD	SE
Linear Regression	33	0.984	0.888	0.433	0.0754
RNN	33	0.749	0.723	0.186	0.0324
RGLM	33	0.635	0.624	0.111	0.0194

According to the estimation method, paired samples t-tests was conducted to see if the magnitude of MASE was significant. The results of paired samples t-tests are shown in Table 5.13.

Table 5.13 Paired samples t-tests result of MASE value

				Mean	SE	95% Confidence Interval		
Pair	t	df	p	difference	difference	Lower	Upper	Cohen's d
Linear Regression - RNN	3.89	32	<.001	0.235	0.0605	0.1328	∞	0.677
Linear Regression - RGLM	5.49	32	<.001	0.349	0.0635	0.2412	∞	0.956
RNN - RGLM	4.71	32	<.001	0.113	0.0241	0.0727	∞	0.821

Note. H_A; Measure 1 >Measure 2

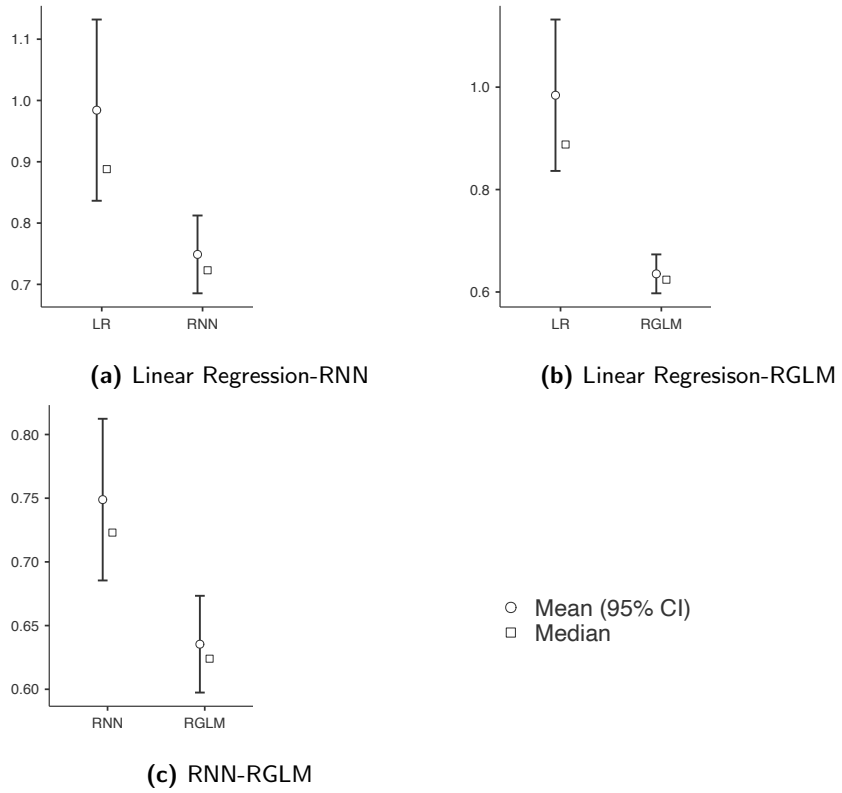
A paired samples t-tests was performed to compare whether the MASE value for the linear regression result was higher than the MASE value for the RNN result. Since the t value is 3.89, the MASE value for the linear regression result is higher than the MASE value for the RNN result under the statistical significance level.

A paired samples t-tests was performed to compare whether the MASE value for the linear regression result was higher than the MASE value for the

RGLM result. Since the t value is 5.49, the MASE value for the linear regression result is higher than the MASE value for the RGLM result under the statistical significance level.

A paired samples t-tests was performed to compare whether the MASE value for the RNN result was higher than the MASE value for the RGLM result. Since the t value is 4.71, the MASE value for the RNN result is higher than the MASE value for the RGLM result under the statistical significance level.

The distribution of 95% confidence intervals and median values for the MASE values for each estimation method is shown in the plot of Figure 5.5.



(a) Linear Regression-RNN

(b) Linear Regresison-RGLM

(c) RNN-RGLM

Figure 5.5 95% Confidence Interval and Median Distribution Chart for MASE Values

5.4.5 Comparison of Root Mean Square Error(RMSE)

Descriptive statistics of RMSE values according to each estimation method can be expressed, as shown in Table 5.14. The RMSE of linear regression is mean 143.5, median 123.0, SD 80.5, SE 14.02. The RMSE of RNN is mean 95.6, median 94.0, SD 18.7, SE 3.26. The RMSE of RGLM is mean 84.6, median 79.0, SD 23.1, SE 4.02.

Table 5.14 Descriptive statistics on RMSE values for each estimation method.

Method	N	Mean	Median	SD	SE
Linear Regression	33	143.5	123	80.5	14.02
RNN	33	95.6	94	18.7	3.26
RGLM	33	84.6	79	23.1	4.02

According to the estimation method, paired samples t-tests was conducted to see if the magnitude of RMSE was significant. The results of paired samples t-tests are shown in Table 5.15.

Table 5.15 Paired samples t-tests result of RMSE value

				Mean	SE	95% Confidence Interval		
Pair	t	df	p	difference	difference	Lower	Upper	Cohen's d
Linear Regression - RNN	3.57	32	$<.001$	47.8	13.41	25.14	∞	0.621
Linear Regression - RGLM	5.27	32	$<.001$	58.9	11.17	39.99	∞	0.918
RNN - RGLM	2.36	32	0.012	11.1	4.69	3.11	∞	0.41

Note. H_A; Measure 1 >Measure 2

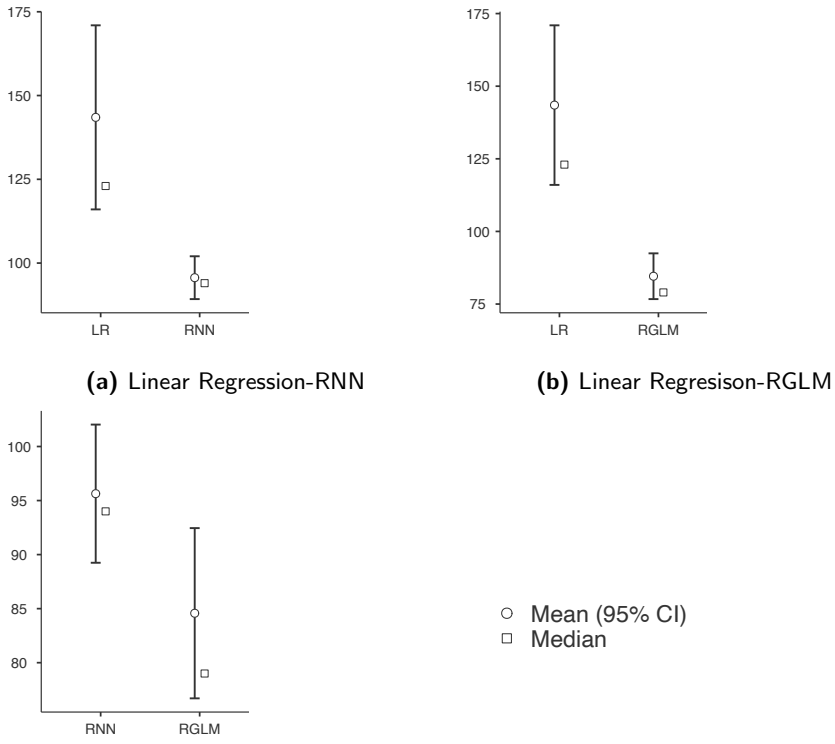
A paired samples t-tests was performed to compare whether the RMSE value for the linear regression result was higher than the RMSE value for the RNN result. Since the t value is 3.57, the RMSE value for the linear regression result is higher than the RMSE value for the RNN result under the statistical significance level.

A paired samples t-tests was performed to compare whether the RMSE value for the linear regression result was higher than the RMSE value for the

RGLM result. Since the t value is 5.27, the RMSE value for the linear regression result is higher than the RMSE value for the RGLM result under the statistical significance level.

A paired samples t-tests was performed to compare whether the RMSE value for the RNN result was higher than the RMSE value for the RGLM result. Since the t value is 2.36, the RMSE value for the RNN result is higher than the RMSE value for the RGLM result under the statistical significance level.

The distribution of 95% confidence intervals and median values for the RMSE values for each estimation method is shown in the plot of Figure 5.6.



(a) Linear Regression-RNN

(b) Linear Regresison-RGLM

(c) RNN-RGLM

Figure 5.6 95% Confidence Interval and Median Distribution Chart for RMSE Values

5.4.6 Comparison of Mean Absolute Error(MAE)

Descriptive statistics of MAE values according to each estimation method can be expressed, as shown in Table 5.16. The MAE of linear regression is mean 100.8, median 96.0, SD 43.2, SE 7.53. The MAE of RNN is mean 76.2, median 74.0, SD 15.9, SE 2.76. The MAE of RGLM is mean 65.5, median 64.0, SD 12.6, SE 2.19.

Table 5.16 Descriptive statistics on MAE values for each estimation method.

Method	N	Mean	Median	SD	SE
Linear Regression	33	100.8	96	43.2	7.53
RNN	33	76.2	74	15.9	2.76
RGLM	33	65.5	64	12.6	2.19

According to the estimation method, paired samples t-tests was conducted to see if the magnitude of MAE was significant. The results of paired samples t-tests are shown in Table 5.17.

Table 5.17 Paired samples t-tests result of MAE value

				Mean	SE	95% Confidence Interval		
Pair	t	df	p	difference	difference	Lower	Upper	Cohen's d
Linear Regression - RNN	3.95	32	$<.001$	24.7	6.24	14.1	∞	0.688
Linear Regression - RGLM	5.54	32	$<.001$	35.3	6.37	24.51	∞	0.965
RNN - RGLM	4.27	32	$<.001$	10.6	2.49	6.42	∞	0.743

Note. H_A; Measure 1 >Measure 2

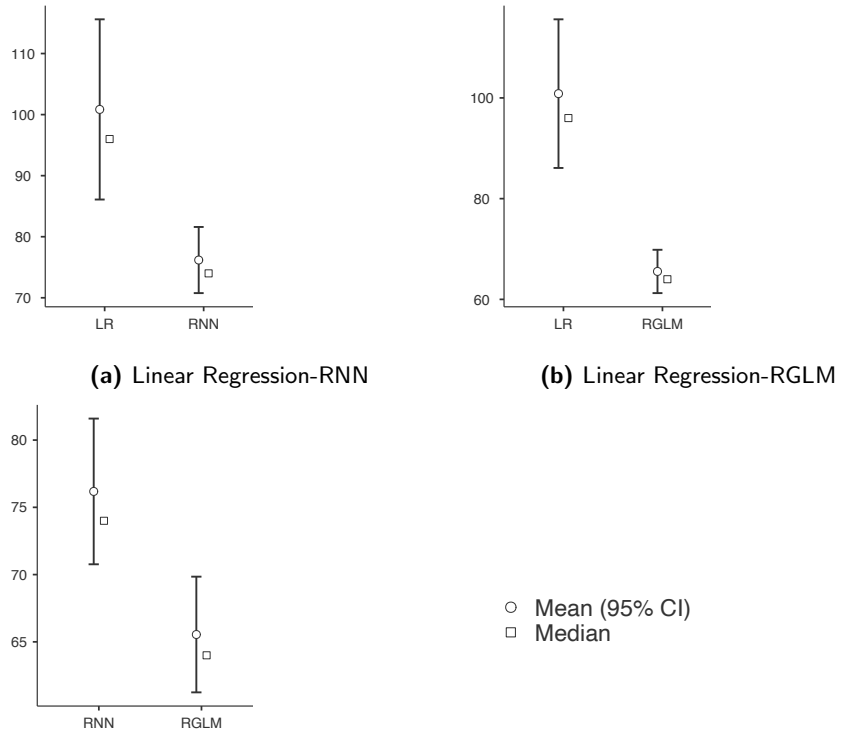
A paired samples t-tests was performed to compare whether the MAE value for the linear regression result was higher than the MAE value for the RNN result. Since the t value is 3.95, the MAE value for the linear regression result is higher than the MAE value for the RNN result under the statistical significance level.

A paired samples t-tests was performed to compare whether the MAE value for the linear regression result was higher than the MAE value for the RGLM

result. Since the t value is 5.54, the MAE value for the linear regression result is higher than the MAE value for the RGLM result under the statistical significance level.

A paired samples t-tests was performed to compare whether the MAE value for the RNN result was higher than the MAE value for the RGLM result. Since the t value is 4.27, the MAE value for the RNN result is higher than the MAE value for the RGLM result under the statistical significance level.

The distribution of 95% confidence intervals and median values for the MAE values for each estimation method is shown in the plot of Figure 5.7.



(a) Linear Regression-RNN

(b) Linear Regression-RGLM

(c) RNN-RGLM

Figure 5.7 95% Confidence Interval and Median Distribution Chart for MAE Values

5.4.7 Comparison of Mean Squared Error(MSE)

Descriptive statistics of MSE values according to each estimation method can be expressed, as shown in Table 5.18. The MSE of linear regression is mean 100.8, median 96.0, SD 43.2, SE 7.53. The MSE of RNN is mean 76.2, median 74.0, SD 15.9, SE 2.76. The MSE of RGLM is mean 65.5, median 64.0, SD 12.6, SE 2.19.

Table 5.18 Descriptive statistics on MSE values for each estimation method.

Method	N	Mean	Median	SD	SE
Linear Regression	33	26861	15245	35446	6170
RNN	33	9501	8812	3632	632
RGLM	33	7671	6225	4515	786

According to the estimation method, paired samples t-tests was conducted to see if the magnitude of MSE was significant. The results of paired samples t-tests are shown in Table 5.19.

Table 5.19 Paired samples t-tests result of MSE value

				Mean	SE	95% Confidence Interval		
Pair	t	df	p	difference	difference	Lower	Upper	Cohen's d
Linear Regression - RNN	2.86	32	0.004	17360	6079	7062	∞	0.497
Linear Regression - RGLM	3.42	32	$<.001$	19191	5614	9681	∞	0.595
RNN - RGLM	1.88	32	0.035	1831	974	181	∞	0.327

Note. H_A; Measure 1 >Measure 2

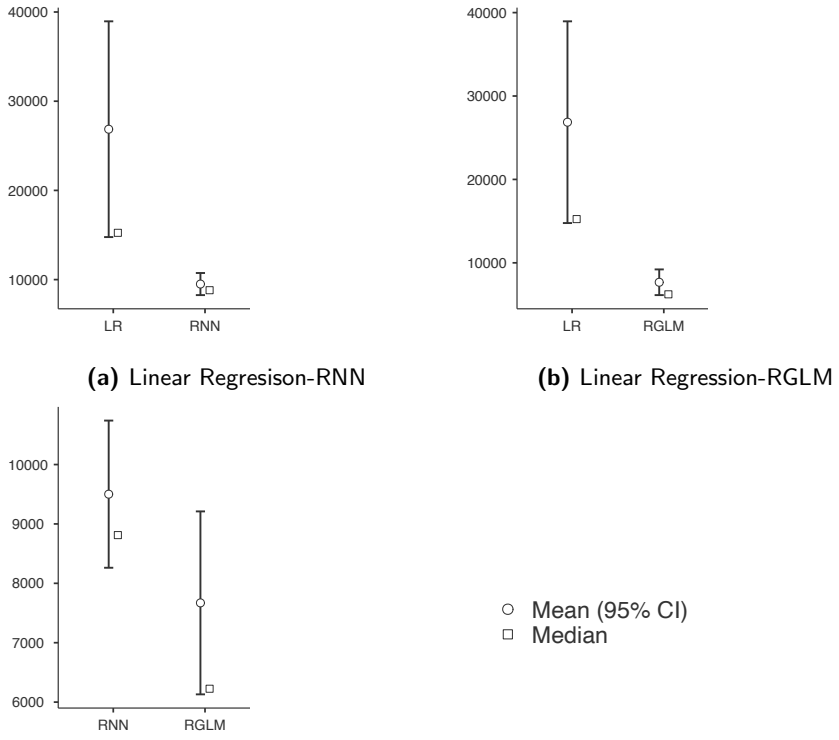
A paired samples t-tests was performed to compare whether the MSE value for the linear regression result was higher than the MSE value for the RNN result. Since the t value is 2.86, the MSE value for the linear regression result is higher than the MSE value for the RNN result under the statistical significance level.

A paired samples t-tests was performed to compare whether the MSE value for the linear regression result was higher than the MSE value for the RGLM

result. Since the t value is 3.42, the MSE value for the linear regression result is higher than the MSE value for the RGLM result under the statistical significance level.

A paired samples t-tests was performed to compare whether the MSE value for the RNN result was higher than the MSE value for the RGLM result. Since the t value is 1.88, the MSE value for the RNN result is higher than the MSE value for the RGLM result under the statistical significance level.

The distribution of 95% confidence intervals and median values for the MSE values for each estimation method is shown in the plot of Figure 5.8.



(a) Linear Regresison-RNN

(b) Linear Regression-RGLM

(c) RNN-RGLM

Figure 5.8 95% Confidence Interval and Median Distribution Chart for MSE Values

5.4.8 Clustering the Results Along with the Performance

Table 5.20 Clustering result of performance comparison by methods

Performance group	Performance order	Hand dimension of stature estimation model
1	Linear Regression >RNN >RGLM	WC, WB, HT, 1DL, 2DL, 4DL, 1D2L, 2D3L, 3D3L, 4D3L, 5D3L, 2D2L, 3D2L, 4D2L, 5D2L, 2D1L, 3D1L, 5D1L, set3
2	Linear Regression >RGLM >RNN	MHB, MHT, HL, MHC, 4D1L, set 1
3	RNN >Linear Regression >RGLM	HC, 3DL, set 2, set 4
exception	-	HB, PL, 5DL, 1D1L

Twenty-nine estimation results can be clustered based on the performance order. Table 5.20 show the list of groups. Group 1 can be classified as having the smallest metric value for RGLM, medium metric value for RNN, and meric value for linear regression. This corresponds to 19 data sets: WC, WB, HT, 1DL, 2DL, 4DL, 1D2L, 2D3L, 3D3L, 4D3L, 5D3L, 2D2L, 3D2L, 4D2L, 5D2L, 2D1L, 3D1L, 5D1L, and Set 3.

Group 2 can be classified as having the smallest metric value for RNN, metric value for RGLM, and meric value for linear regression. This corresponds to six data sets: MHB, MHT, HL, MHC, 4D1L, and Set 1.

Group 3 can be classified as having the smallest metric value for RGLM, medium metric value for linear regression, and meric value for RNN. This corresponds to four data sets: HC, 3DL, Set 2, and Set 4.

Of the 33 data sets, there were 4 cases of exceptional stature estimation results. HB, PL, 5DL, and 1D1L showed different performance order according to metric.

5.5 Discussion

Linear regression, RNN, and RGLM were used to estimate the stature from the 29 part sizes of the hand. As a result, 99 models were created to assess the stature. MASE, RMSE, MAE, MSE, RAE, RSE, and MAPE were used as metrics to evaluate the estimated performance of each of these models.

To see if there is a difference in the RAE values according to the estimation methods, RAE of linear regression > RAE of RNN, RAE of linear regression > RAE of RGLM, and RAE of RNN > RAE of RGLM were hypothesized, and paired sampels t-tests were performed. As a result, RAE of linear regression was significantly larger than RAE of RNN (t=3.94, p<.001). RAE of linear regression was significantly higher than RAE of RGLM (t=5.81, p<.001). RAE of RNN was significantly larger than RAE of RGLM (t=4.53, p<.001). The average comparison result is RAE of linear regression (Mean=1.329) > RAE of RNN (Mean=1.012) > RAE of RGLM (Mean = 0.83). When comparing the estimated performance based on the RAE value, RGLM was the best, followed by RNN, followed by linear regression.

To see if there is a difference in the RSE values according to the estimation methods, RSE of linear regression > RSE of RNN, RSE of linear regression > RSE of RGLM, and RSE of RNN > RSE of RGLM were hypothesized, and paired samples t-tests were performed. As a result, RSE of linear regression was significantly larger than RSE of RNN (t=2.77, p=0.005). RSE of linear regression was significantly larger than RSE of RGLM (t=3.41, p<.001). Although the RSE of RNN had overlapping regions in the 95% confidence interval with the RSE of RGLM, the RSE of RNN was more significant than the RSE of RGLM (t=2.17, p=0.019). The average comparison result is RSE of linear regression (Mean=3.186) > RSE of RNN (Mean=1.197) > RSE of RGLM (Mean=0.933).

When comparing the estimated performance based on the RSE values, RGLM was the best, RNN was the next, and linear regression was the worst.

MAPE and MAE showed the same pattern as RAE, while RMSE and MSE showed the same pattern as RSE. RAE, MAPE, and MASE are evaluation metric using the absolute value of the difference between the actual value and the predicted value, and RSE, RMSE, and MSE use the square of the difference between the actual value and the predicted value.

RGLM is the best when comparing the estimation performance for the overall estimation results, followed by RNN, linear regression. This is shown in 19 datasets out of 33 datasets. This estimated performance order is clusteded as Group 1. In Group 2, the performance of RNN was the best for six datasets(MHB, MHT, HL, MHC, 4D1L, set 1), followed by RGLM and linear regression. Group 3 can be classified as having the smallest metric value for RGLM, medium metric value for linear regression, and the biggest metric value for RNN. This corresponds to four data sets: HC, 3DL, Set 2, and Set 4. The remaining four datasets showed different performance evaluation results according to the evaluation metric.

In every group, RGLM has always performed better than linear regression. In Group 3, RNN's estimation performance was the worst, while in Group 2, RNN's estimation performance was the best. Of the four datasets clustered as exceptions, 5DL and 1D1L have a better estimation of RGLM than linear regression.

When the stature was estimated from HB, the performance of RNN was the lowest when the estimation performance was estimated by metric using absolute error, and the performance of RGLM was the lowest when the estimation performance was evaluated by metric using squared error. Since results may

vary depending on the evaluation metric, it is not appropriate to use RNN and RGLM when estimating stature from HB.

Estimating stature from the size of body parts is meaningful in the forensic domain. Jee and Yun (2015) used linear regression to estimate stature from the hand dimensions. In this research, stature estimation via RNN and RGLM, which are the kind of deep-learning, showed better estimation performance than linear regression. In some dimensions such as HB and PL, the performance of linear regression is better than others. However, for other dimensions of the hand, the performance of RNN and RGLM is much better than linear regression. In the case of RNN, the estimation performance was worse than that of linear regression in Group 3. However, RGLM showed better estimation performance than linear regression in most cases. Therefore, in estimating stature in actual forensics, RGLM can be said to have more practical meaning.

Chapter 6

Discussion and Conclusions

6.1 Summary of findings

This study explores the possibility of using deep-learning in various kinds of ergonomic research data such as numerical, image, and video. As the image data, the body pressure distribution image of the seat plate according to the posture was used. The video data covered in this study are repetitive work images of factory workers. The numerical data covered in this study are the human body size data collected by the Size Korea project. This study aimed to derive a deep-learning methodology suitable for the types of data and the kinds of results to be estimated. The findings of this study can be summarized below.

The posture was classified via CNN to estimate the sitting posture from the body pressure distribution image. Cross-validation was performed to eliminate the effects of dataset combination. As a result, the accuracy was at least 0.358, and at most, 0.826, which did not show outstanding accuracy. However,

compared to logistic regression, there was a 20% improvement. Sitting straight, lean forward, and lean backward are estimated with less than half accuracy. In this regard, the body pressure distribution of the body tilt in the forward and backward directions is similar. So, the classification between these three postures was not well done, so the overall accuracy was low. Although it used LeNet-5, an early CNN algorithm with relatively low performance, it showed better performance than logistic regression. It showed the possibility of applying deep learning technology to everyday products.

In Chapter 4, the work video is analyzed via LSTM to estimate the work risk assessment. Since OWAS, RULA, and REBA are numerical evaluation tools of integer type, the estimation model is composed of the classification model, not a regression model. The accuracy of OWAS is 0.531, the accuracy of RULA is 0.241, and the accuracy of REBA is 0.184. The classification result was not good because training data was created without enough data for each case to classify. If training data is biased or missing in a specific case, test data cannot be appropriately classified. In this study, the training data did not cover all cases, so the overall classification performance was low. In the case of OWAS, four action levels were classified, so performance was relatively good. If the training data includes the entire case evenly, good results are expected.

In Chapter 5, RNN and RGLM are applied to estimate the stature from the hand dimensions. The performance of linear regression, RNN, and RGLM used to estimate the stature from each hand dimension, and the hand dimension combinations were compared. RAE, RSE, MAPE, MASE, RMSE, MAE, and MSE were used as metrics to evaluate regression performance. These evaluation metrics are classified into two categories. It is a metric using the absolute value of the difference between the predicted value and the actual value, and a metric

using the square value of the difference between the predicted value and the actual value. For the same group of metrics, the values are different, but the performance order is the same. Instead, there were cases where the performance order differed between groups. When comparing the estimated performance, the performance of RNN and RGLM was better than linear regression. Through paired sample t-test, the estimated performance of RGLM was better than that of RNN. However, depending on the kind of hand dimension, the performance of RNN was better than RGLM. Therefore, when comparing the performance of the regression model, it is necessary to apply both a metric using absolute value and a metric using a square value.

6.2 Contributions of this study

In this study, the method of ergonomic analysis via deep-learning was derived. As technology advances, the kinds of data that ergonomics deal with are more diverse. The kinds and amounts of data to be analyzed, such as various sensor data collected in real-time and video data recorded through CCTV installed around us, have increased. As various sensor data are collected in real-time and video data is continuously collected through CCTV, the kinds and amounts of data to be analyzed have increased. These data are plentiful and diverse for ergonomic researchers to analyze conventionally. This study contributes to that it has been confirmed that deep-learning is a suitable method for processing such data.

Body pressure distribution image data was used to derive a method for analyzing sensor data collected in real-time through deep-learning. Since two body pressure distribution data are collected every second, it is almost impossible for the researcher to analyze them in real-time. However, the deep-learning model

built in this study enabled real-time posture estimation from body pressure distribution image data. The modified LeNet-5 used in this study classified seven postures with an accuracy of 0.616. The logistic regression classified seven postures with an accuracy of 0.507. Although LeNet-5 was an early CNN method, it showed better performance than logistic regression. This study showed an accuracy of 0.616, even though the early CNN LeNet-5 was used. It is not the latest deep-learning model, but it showed that excellent performance is a technique applicable to everyday products.

In this study, training data and test data do not overlap. This situation is similar to the user experience of a product with a deep-learning model applied. The deep-learning model tends to increase in accuracy as the size of training data increases (Cho et al., 2015). However, the model cannot be improved immediately by adding new data. Training is required for data that combines existing data with new data. First-time users of products with deep-learning applied are likely to use products that do not reflect their data. In this study, it was significant that the posture classification performance was excellent even though the training data and test data did not overlap.

The motion of the worker was extracted from the video recording the automobile assembly task. Work risk assessment was performed through LSTM, a kind of RNN, with the extracted worker's movement. When OWAS, RULA, and REBA were estimated using LSTM, the performance was not excellent. The accuracy of the OWAS value estimation result is 0.531. The accuracy of the RULA value estimation result is 0.241, and the accuracy of the REBA value estimation result is 0.184. It is premature to analyze work videos by deep-learning to do a work risk assessment. However, the OpenPose library, a deep-learning library used to extract worker movements, can be considered suitable for worker

movement extraction. If the cases corresponding to each work risk level can be collected as training data without bias, deep-learning can be used for work risk assessment.

RNN and RGLM are used to estimate the stature from the hand dimensions. Models for estimating stature using RNN have better estimation performance than those for estimating stature using linear regression. Models for estimating stature using RLGM performed better than models for estimating stature using RNN. Therefore, RGLM is a methodology that can estimate the stature from hand dimensions closer to the actual stature than other methods. It will be an easy way to estimate the stature in situations where it is difficult to measure the stature, such as criminal investigations.

Seven evaluation metrics were used to measure regression performance. In the case of RAE, MAPE, MASE, and MAE, which use the absolute value of the difference between the actual value the predicted value, the values were different, but the performance order was not changed. Similarly, RSE, RMSE, and MSE, which use the square of the difference between the actual and predicted values, also differ in value, but not in the case. There have been cases where the performance order between metrics using absolute value and metrics using square value is different. There are various regression performance evaluation metrics, but two metrics are sufficient to make relative comparisons of performance. One is a metric that uses the square of the difference between the actual and predicted values. The other is a metric that uses the absolute value of the difference between the actual and predicted values.

Deep-learning techniques are not only one but various techniques. They can be applied or combined depending on the nature of the data. Deep-learning can be applied directly to real data. However, deep-learning can also be ap-

plied as a preprocess to individual data. OpenPose library, which is used to extract human body motions from factory workers' work images, is also a motion extraction library developed through deep-learning. In this way, the result obtained through deep-learning can be reused to obtain another result.

Ergonomic research using deep-learning does not mean that AI replaces researchers. Deep-learning is a means of expanding the breadth of data available to researchers. Ergonomic know-how is essential for labeling to create training data. These deep-learning models will contribute to the automation of ergonomic analysis.

6.3 Limitations and further studies

In this study, various attempts were made to analyze ergonomic data through deep learning, and there were limitations.

When estimating the sitting posture, there was an up-to-date, high-performance algorithm, but LeNet-5 was used because it does not require many resources. It would be expected that improvements to accuracy would have been possible using the latest image classification techniques. Since the post-test on individual data was not conducted, both the training data and the test data contained many noise data. However, it is natural to include noise data if the test is performed under the assumption of the actual use environment. It would be meaningful to study the average noise data rate of sensor data collected in everyday life.

The accuracy was very low when OWAS, RULA, and REBA scores were estimated. The data was not collected for deep-learning, and there were not enough cases for every assessment score. Furthermore, because each video's

angle is not unified, the body parts displayed on the screen may vary depending on the video. OpenPose does not show 3D information and extracts motion in 2D. If 3D information can be extracted from the image, a more accurate work risk assessment can be performed.

When estimating stature using the hand dimensions, this study did not consider gender. If hand dimensions and gender information are considered together, the stature can be estimated more accurately.

Bibliography

Aboul-Hagag, K. E., Mohamed, S. A., Hilal, M. A., and Mohamed, E. A. (2011). Determination of sex from hand dimensions and index/ring finger length ratio in upper egyptians. *Egyptian Journal of Forensic Sciences*, 1(2):80–86.

Agnihotri, A. K., Agnihotri, S., Jeebun, N., and Googoolye, K. (2008). Prediction of stature using hand dimensions. *Journal of forensic and legal medicine*, 15(8):479–482.

Ahmed, A. A. (2013). Estimation of stature from the upper limb measurements of sudanese adults. *Forensic science international*, 228(1-3):178.e1–178.e7.

Akhlaghi, M., Hajibeygi, M., Zamani, N., and Moradi, B. (2012). Estimation of stature from upper limb anthropometry in iranian population. *Journal of forensic and legal medicine*, 19(5):280–284.

Andersson, B. and Ortengren, R. (1974). Lumbar disc pressure and myoelectric back muscle activity during sitting. i. studies on an experimental chair. *Scandinavian Journal of rehabilitation medicine*, 6(3):104.

Bao, J., Li, W., Li, J., Ge, Y., and Bao, C. (2013). Sitting posture recognition based on data fusion on pressure cushion. *Indonesian Journal of Electrical Engineering and Computer Science*, 11(4):1769–1775.

Barba, R., de Madrid, P., and Boticario, J. G. (2015). Development of an inexpensive sensor network for recognition of sitting posture. *International Journal of Distributed Sensor Networks*, 11(8):969237.

Bendix, T. and Biering-SÃžrensen, F. (1983). Posture of the trunk when sitting on forward inclining seats. *Scandinavian journal of rehabilitation medicine*, 15(4):197–203.

Benocci, M., Farella, E., and Benini, L. (2011). A context-aware smart seat. In *Advances in Sensors and Interfaces (IWASI), 2011 4th IEEE International Workshop on*, pages 104–109. IEEE.

Boulay, B., Bremond, F., and Thonnat, M. (2005). Posture recognition with a 3d human model. In *Imaging for Crime Detection and Prevention, 2005. ICDP 2005. The IEE International Symposium on*, pages 135–138. IET.

Brolin, E., Högberg, D., Hanson, L., and Örtengren, R. (2017). Adaptive regression model for synthesizing anthropometric population data. *International Journal of Industrial Ergonomics*, 59:46–53.

Carnahan, B. J. and Redfern, M. S. (1998a). Application of genetic algorithms to the design of lifting tasks. *International Journal of Industrial Ergonomics*, 21(2):145–158.

Carnahan, B. J. and Redfern, M. S. (1998b). Building a low back injury risk classifier using evolutionary computation. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 42, pages 881–885. SAGE Publications Sage CA: Los Angeles, CA.

Chaffin, D. B., Andersson, G. B., and Martin, B. J. (2006). *Occupational biomechanics*. John Wiley & Sons.

Cho, J., Lee, K., Shin, E., Choy, G., and Do, S. (2015). Medical image deep learning with hospital PACS dataset. *CoRR*, abs/1511.06348.

Choi, H. S., Yi, K. H., Kang, Y. S., and Kim, E. K. (2006). Development of hand part measurement protocol for glove design. Technical report, Ministry of Commerce Industry and Energy.

Dendamrongvit, T. (2002). Development of a posture prediction model. Master's thesis, Oregon State University.

Dunne, L. E., Walsh, P., Hermann, S., Smyth, B., and Caulfield, B. (2008). Wearable monitoring of seated spinal posture. *IEEE transactions on biomedical circuits and systems*, 2(2):97–105.

Dunstan, D., Barr, E., Healy, G., Salmon, J., Shaw, J., Balkau, B., Magliano, D., Cameron, A., Zimmet, P., and Owen, N. (2010). Television viewing time and mortality the australian diabetes, obesity and lifestyle study (ausdiab). *Circulation*, 121(3):384–391.

Fard, F. D., Moghimi, S., and Lotfi, R. (2013). Evaluating pressure ulcer development in wheelchair-bound population using sitting posture identification. *Engineering*, 5(10):132.

Foerster, F., Smeja, M., and Fahrenberg, J. (1999). Detection of posture and motion by accelerometry: a validation study in ambulatory monitoring. *Computers in Human Behavior*, 15(5):571–583.

Gers, F. A., Schmidhuber, J., and Cummins, F. (1999). Learning to forget: Continual prediction with lstm. *Neural Comput.*

Habib, S. R. and Kamal, N. N. (2010). Stature estimation from hand and phalanges lengths of egyptians. *Journal of forensic and legal medicine*, 17(3):156–160.

Hadler, N. M., Gillings, D. B., Imbus, H. R., Levitin, P. M., Makuc, D., Utsinger, P. D., Yount, W. J., Slusser, D., and Moskovitz, N. (1978). Hand structure and function in an industrial setting. *Arthritis & Rheumatism: Official Journal of the American College of Rheumatology*, 21(2):210–220.

Haslegrave, C. M. (1994). What do we mean by a 'working posture'. *Ergonomics*, 37(4):781–799.

Healy, G. N., Wijndaele, K., Dunstan, D. W., Shaw, J. E., Salmon, J., Zimmet, P. Z., and Owen, N. (2008). Objectively measured sedentary time, physical activity, and metabolic risk the australian diabetes, obesity and lifestyle study (ausdiab). *Diabetes care*, 31(2):369–371.

Hedge, A. (2000a). Reba employee assessment worksheet. *Cornell University*.

Hedge, A. (2000b). Rula employee assessment worksheet. *Cornell University*.

Herron, R. E. (2000). Anthropometry: Definition, uses and methods of measurement. *International Encyclopedia of Ergonomics and Human Factors-3 Volume Set*, page 879.

Hu, Y., Stoelting, A., Wang, Y.-T., Zou, Y., and Sarrafzadeh, M. (2010). Providing a cushion for wireless healthcare application development. *IEEE potentials*, 29(1):19–23.

Huberty, C. J. and Lowman, L. L. (1997). Discriminant analysis via statistical packages. *Educational and Psychological Measurement*, 57(5):759–784.

Ishak, N.-I., Hemy, N., and Franklin, D. (2012). Estimation of stature from hand and handprint dimensions in a western australian population. *Forensic science international*, 216(1-3):199.e1–199.e7.

Jee, S.-c. and Yun, M. H. (2015). Estimation of stature from diversified hand anthropometric dimensions from korean population. *Journal of forensic and legal medicine*, 35:9–14.

Juul-Kristensen, B., Hansson, G.-Å., Fallentin, N., Andersen, J., and Ekdahl, C. (2001). Assessment of work postures and movements using a video-based observation method and direct technical measurements. *Applied ergonomics*, 32(5):517–524.

Kamiya, K., Kudo, M., Nonaka, H., and Toyama, J. (2008). Sitting posture analysis by pressure sensors. In *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pages 1–4. IEEE.

Kanchan, T. and Krishan, K. (2011). Anthropometry of hand in sex determination of dismembered remains - a review of literature. *Journal of Forensic and Legal Medicine*, 18(1):14–17.

Kanchan, T., Krishan, K., Shyamsundar, S., Aparna, K., and Jaiswal, S. (2012). Analysis of footprint and its parts for stature estimation in indian population. *The foot*, 22(3):175–180.

Kanchan, T. and Rastogi, P. (2009). Sex determination from hand dimensions of north and south indians. *Journal of forensic sciences*, 54(3):546–550.

Karhu, O., Kansi, P., and Kuorinka, I. (1977). Correcting working postures in industry: a practical method for analysis. *Applied ergonomics*, 8(4):199–201.

Koza, J. R., Bennett, F., Andre, D., and Keane, M. A. (1999). Genetic programming iii: darwinian invention and problem solving [book review]. *IEEE Transactions on Evolutionary Computation*, 3(3):251–253.

Le, X.-H., Ho, H. V., Lee, G., and Jung, S. (2019). Application of long short-term memory (lstm) neural network for flood forecasting. *Water*, 11(7):1387.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lin, S., Cai, L., Lin, X., and Ji, R. (2016). Masked face detection via a modified lenet. *Neurocomputing*, 218:197–202.

Luopajarvi, T. (1990). Ergonomic analysis of workplace and postural load. *Ergonomics: the physiotherapist in the workplace. Edinburgh, London, Melbourne and New York*, pages 51–78.

Ma, C., Li, W., Gravina, R., and Fortino, G. (2016). Activity recognition and monitoring for smart wheelchair users. In *Computer Supported Cooperative Work in Design (CSCWD), 2016 IEEE 20th International Conference on*, pages 664–669. IEEE.

Marmaras, N., Poulakakis, G., and Papakostopoulos, V. (1999). Ergonomic design in ancient Greece. *Applied Ergonomics*, 30(4):361.

McAtamney, L. and Corlett, E. N. (1993). Rula: a survey method for the investigation of work-related upper limb disorders. *Applied ergonomics*, 24(2):91–99.

McAtamney, L. and Hignett, S. (2000). Reba: Rapid entire body assessment. *Applied ergonomics*, 31:201–205.

Meyer, J., Arnrich, B., Schumm, J., and Troster, G. (2010). Design and modeling of a textile pressure sensor for sitting posture classification. *IEEE Sensors Journal*, 10(8):1391–1398.

Mota, S. and Picard, R. W. (2003). Automated posture analysis for detecting learner's interest level. In *Computer Vision and Pattern Recognition Workshop, 2003. CVPRW'03. Conference on*, volume 5, pages 49–49. IEEE.

Mutlu, B., Krause, A., Forlizzi, J., Guestrin, C., and Hodgins, J. (2007). Robust, low-cost, non-intrusive sensing and recognition of seated postures. In *Proceedings of the 20th annual ACM symposium on User interface software and technology*, pages 149–158. ACM.

Myers, R. (1990). Traditional and modern regression with applications.

O'sullivan, P. B., Grahamslaw, K. M., Kendell, M., Lapenskie, S. C., Möller, N. E., and Richards, K. V. (2002). The effect of different standing and sitting postures on trunk muscle activity in a pain-free population. *Spine*, 27(11):1238–1244.

Pham, D. and Onder, H. (1992). A knowledge-based system for optimizing workplace layouts using a genetic algorithm. *Ergonomics*, 35(12):1479–1487.

Rohmert, W. and Mainzer, J. (1986). Influence and assessment methods for evaluating body postures. *The ergonomics of working postures. London: Taylor and Francis*, pages 183–217.

Sarvadevabhatla, R. K. and Babu, R. V. (2015). Freehand sketch recognition using deep features. *arXiv preprint arXiv:1502.00254*.

Sohn, S. Y. and Shin, H. (2001). Pattern recognition for road traffic accident severity in korea. *Ergonomics*, 44(1):107–117.

Tan, H. Z., Slivovsky, L. A., and Pentland, A. (2001). A sensing chair using pressure distribution sensors. *IEEE/ASME Transactions On Mechatronics*, 6(3):261–268.

Taylor, F. W. (1903). Time study, piece work, and the first-class man. *HF Merrill (Ed.)*.

Uhrová, P., Beňuš, R., and Masnicová, S. (2013). Stature estimation from various foot dimensions among slovak population. *Journal of forensic sciences*, 58(2):448–451.

Videman, T., Nurminen, M., and Troup, J. (1990). 1990 volvo award in clinical sciences. lumbar spinal pathology in cadaveric material in relation to history of back pain, occupation, and physical loading. *Spine*, 15(8):728–740.

Xu, L., Chen, G., Wang, J., Shen, R., and Zhao, S. (2012). A sensing cushion using simple pressure distribution sensors. In *Multisensor Fusion and Integration for Intelligent Systems (MFI), 2012 IEEE Conference on*, pages 451–456. IEEE.

Xu, W., Huang, M.-C., Amini, N., He, L., and Sarrafzadeh, M. (2013). ecushion: A textile pressure sensor array design and calibration for sitting posture analysis. *IEEE Sensors Journal*, 13(10):3926–3934.

Yen, T. Y. and Radwin, R. G. (2002). A comparison between analysis time and inter-analyst reliability using spectral analysis of kinematic data and posture classification. *Applied Ergonomics*, 33(1):85–93.

Zemp, R., Tanadini, M., Plüss, S., Schnüriger, K., Singh, N. B., Taylor, W. R., and Lorenzetti, S. (2016). Application of machine learning approaches for classifying sitting posture based on force and acceleration sensors. *BioMed research international*, 2016.

Appendix A

Confusion Matrix from Chapter III

Table A.1 Logistic Regression, Cross-validation No. 1

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	66	97	1	0	14	0	82	260	0.25
b	151	109	0	0	0	0	0	260	0.42
c	0	107	113	0	3	0	37	260	0.43
d	3	34	0	223	0	0	0	260	0.86
e	59	40	0	0	158	0	3	260	0.61
f	0	0	0	0	0	240	20	260	0.92
g	58	1	0	1	64	0	136	260	0.52
Total	337	388	114	224	239	240	278	Accuracy	
Precision	0.20	0.28	0.99	1.00	0.66	1.00	0.49	0.57	

Table A.2 CNN, Cross-validation No. 1

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	46	63	2	22	127	0	0	260	0.18
b	50	111	0	7	90	2	0	260	0.43
c	0	110	114	0	34	2	0	260	0.44
d	0	1	0	227	32	0	0	260	0.87
e	39	2	1	0	209	9	0	260	0.80
f	0	0	0	0	0	260	0	260	1
g	0	0	2	2	1	62	193	260	0.74
Total	135	287	119	258	493	335	193	Accuracy	
Precision	0.34	0.39	0.96	0.88	0.42	0.78	1	0.64	

Table A.3 Logistic Regression, Cross-validation No. 2

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	0	0	4	256	0	0	0	260	0.00
b	0	0	3	196	0	0	61	260	0.00
c	0	0	259	1	0	0	0	260	1.00
d	0	0	0	260	0	0	0	260	1.00
e	0	0	2	258	0	0	0	260	0.00
f	0	0	40	173	0	47	0	260	0.18
g	0	0	63	168	0	0	29	260	0.11
Total	0	0	371	1312	0	47	90	Accuracy	
Precision	-	-	0.70	0.20	-	1.00	0.32	0.33	

Table A.4 CNN, Cross-validation No. 2

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	58	84	4	99	15	0	0	260	0.22
b	33	190	0	37	0	0	0	260	0.73
c	1	10	246	0	2	0	1	260	0.95
d	0	89	0	118	36	17	0	260	0.45
e	2	118	1	38	101	0	0	260	0.39
f	0	18	41	0	9	192	0	260	0.74
g	37	23	10	22	0	0	168	260	0.65
Total	131	532	302	314	163	209	169	Accuracy	
Precision	0.44	0.36	0.81	0.38	0.62	0.92	0.99	0.59	

Table A.5 Logistic Regression, Cross-validation No. 3

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	0	242	0	18	0	0	0	260	0.00
b	0	249	11	0	0	0	0	260	0.96
c	0	196	64	0	0	0	0	260	0.25
d	0	4	0	256	0	0	0	260	0.98
e	0	174	0	66	20	0	0	260	0.08
f	0	260	0	0	0	0	0	260	0.00
g	0	0	0	0	0	0	260	260	1.00
Total	0	1125	75	340	20	0	260	Accuracy	
Precision	-	0.22	0.85	0.75	1.00	-	1.00	0.47	

Table A.6 CNN, Cross-validation No. 3

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	144	0	0	116	0	0	0	260	0.55
b	119	71	42	21	7	0	0	260	0.27
c	60	44	147	0	7	0	2	260	0.57
d	0	0	0	260	0	0	0	260	1
e	108	51	0	61	36	0	4	260	0.14
f	0	10	0	0	0	250	0	260	0.96
g	0	0	1	0	0	0	259	260	1.00
Total	431	176	190	458	50	250	265	Accuracy	
Precision	0.33	0.40	0.77	0.57	0.72	1	0.98	0.64	

Table A.7 Logistic Regression, Cross-validation No. 4

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	153	0	48	47	0	11	1	260	0.59
b	29	0	112	71	0	48	0	260	0.00
c	0	0	255	0	0	5	0	260	0.98
d	13	0	16	193	0	38	0	260	0.74
e	24	0	37	42	0	156	1	260	0.00
f	0	0	22	0	0	229	9	260	0.88
g	1	0	3	1	0	0	255	260	0.98
Total	220	0	493	354	0	487	266	Accuracy	
Precision	0.70	-	0.52	0.55	-	0.47	0.96	0.60	

Table A.8 CNN, Cross-validation No. 4

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	108	32	1	38	81	0	0	260	0.42
b	0	99	0	65	95	0	1	260	0.38
c	0	4	181	0	72	1	2	260	0.70
d	0	1	0	218	25	13	3	260	0.84
e	4	19	2	46	185	4	0	260	0.71
f	0	7	1	3	3	245	1	260	0.94
g	0	66	0	4	32	0	158	260	0.61
Total	112	228	185	374	493	263	165	Accuracy	
Precision	0.96	0.43	0.98	0.58	0.38	0.93	0.96	0.66	

Table A.9 Logistic Regression, Cross-validation No. 5

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	0	0	0	0	260	0	0	260	0.00
b	0	0	19	1	166	4	70	260	0.00
c	0	0	199	0	61	0	0	260	0.77
d	0	0	0	152	90	0	18	260	0.58
e	0	0	0	3	257	0	0	260	0.99
f	0	0	0	0	0	260	0	260	1.00
g	0	0	2	8	38	0	212	260	0.82
Total	0	0	220	164	872	264	300	Accuracy	
Precision	-	-	0.90	0.93	0.29	0.98	0.71	0.59	

Table A.10 CNN, Cross-validation No. 5

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	21	0	0	176	63	0	0	260	0.08
b	0	33	0	90	120	17	0	260	0.13
c	0	3	195	0	62	0	0	260	0.75
d	1	0	0	188	71	0	0	260	0.72
e	7	1	0	128	124	0	0	260	0.48
f	0	0	0	0	2	258	0	260	0.99
g	0	0	1	1	61	0	197	260	0.76
Total	29	37	196	583	503	275	197	Accuracy	
Precision	0.72	0.89	0.99	0.32	0.25	0.94	1	0.56	

Table A.11 Logistic Regression, Cross-validation No. 6

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	0	130	0	62	68	0	0	260	0.00
b	0	130	0	59	71	0	0	260	0.50
c	0	2	243	0	15	0	0	260	0.93
d	0	0	0	260	0	0	0	260	1.00
e	0	10	0	72	178	0	0	260	0.68
f	0	0	0	0	0	260	0	260	1.00
g	0	0	2	0	14	0	244	260	0.94
Total	0	272	245	453	346	260	244	Accuracy	
Precision	-	0.48	0.99	0.57	0.51	1.00	1.00	0.72	

Table A.12 CNN, Cross-validation No. 6

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	27	0	0	65	168	0	0	260	0.10
b	53	106	0	40	61	0	0	260	0.41
c	0	0	260	0	0	0	0	260	1
d	0	1	0	259	0	0	0	260	1.00
e	2	6	0	1	248	2	1	260	0.95
f	0	0	0	0	0	260	0	260	1
g	0	0	64	0	0	0	196	260	0.75
Total	82	113	324	365	477	262	197	Accuracy	
Precision	0.33	0.94	0.80	0.71	0.52	0.99	0.99	0.75	

Table A.13 Logistic Regression, Cross-validation No. 7

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	0	0	0	0	260	0	0	260	0.00
b	0	0	3	0	257	0	0	260	0.00
c	0	0	257	0	3	0	0	260	0.99
d	0	0	0	104	156	0	0	260	0.40
e	0	0	0	0	260	0	0	260	1.00
f	0	0	0	0	85	175	0	260	0.67
g	0	0	1	0	89	0	170	260	0.65
Total	0	0	261	104	1110	175	170	Accuracy	
Precision	-	-	0.98	1.00	0.23	1.00	1.00	0.53	

Table A.14 CNN, Cross-validation No. 7

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	163	55	0	0	39	3	0	260	0.63
b	75	136	45	0	4	0	0	260	0.52
c	3	0	257	0	0	0	0	260	0.99
d	0	0	0	259	0	0	1	260	1.00
e	155	29	22	1	52	0	1	260	0.2
f	0	0	0	0	0	260	0	260	1
g	0	0	0	0	0	0	260	260	1
Total	396	220	324	260	95	263	262	Accuracy	
Precision	0.41	0.62	0.79	1.00	0.55	0.99	0.99	0.76	

Table A.15 Logistic Regression, Cross-validation No. 8

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	88	170	2	0	0	0	0	260	0.34
b	10	220	0	30	0	0	0	260	0.85
c	22	133	105	0	0	0	0	260	0.40
d	50	123	9	78	0	0	0	260	0.30
e	37	214	8	1	0	0	0	260	0.00
f	2	257	0	0	0	1	0	260	0.00
g	19	110	4	2	0	0	125	260	0.48
Total	228	1227	128	111	0	1	125	Accuracy	
Precision	0.39	0.18	0.82	0.70	-	1.00	1.00	0.34	

Table A.16 CNN, Cross-validation No. 8

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	131	2	43	3	58	0	23	260	0.50
b	46	74	3	35	38	46	18	260	0.28
c	10	1	134	1	75	0	39	260	0.52
d	73	6	15	104	34	0	28	260	0.4
e	91	12	50	9	93	2	3	260	0.36
f	0	0	0	3	5	251	1	260	0.97
g	0	0	0	1	0	0	259	260	1.00
Total	351	95	245	156	303	299	371	Accuracy	
Precision	0.37	0.78	0.55	0.67	0.31	0.84	0.70	0.57	

Table A.17 Logistic Regression, Cross-validation No. 9

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	0	260	0	0	0	0	0	260	0.00
b	0	255	5	0	0	0	0	260	0.98
c	0	2	258	0	0	0	0	260	0.99
d	0	174	0	86	0	0	0	260	0.33
e	0	260	0	0	0	0	0	260	0.00
f	0	178	36	0	0	45	1	260	0.17
g	0	76	4	0	0	0	180	260	0.69
Total	0	1205	303	86	0	45	181	Accuracy	
Precision	-	0.21	0.85	1.00	-	1.00	0.99	0.45	

Table A.18 CNN, Cross-validation No. 9

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	2	24	0	154	80	0	0	260	0.01
b	13	76	0	23	138	10	0	260	0.29
c	0	37	202	0	21	0	0	260	0.78
d	0	3	0	180	76	0	1	260	0.69
e	0	61	0	48	140	0	11	260	0.54
f	0	0	0	0	1	259	0	260	1.00
g	0	4	0	24	4	0	228	260	0.88
Total	15	205	202	429	460	269	240	Accuracy	
Precision	0.13	0.37	1	0.42	0.30	0.96	0.95	0.60	

Table A.19 Logistic Regression, Cross-validation No. 10

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	202	0	5	0	0	0	53	260	0.78
b	195	0	15	10	0	0	40	260	0.00
c	65	0	130	0	0	0	65	260	0.50
d	57	0	0	202	0	0	1	260	0.78
e	165	0	1	0	0	1	93	260	0.00
f	0	0	0	1	0	259	0	260	1.00
g	2	0	1	14	0	0	243	260	0.93
Total	686	0	152	227	0	260	495	Accuracy	
Precision	0.29	-	0.86	0.89	-	1.00	0.49	0.57	

Table A.20 CNN, Cross-validation No. 10

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	19	3	20	6	209	0	3	260	0.07
b	28	78	61	11	25	0	57	260	0.3
c	1	51	181	0	24	0	3	260	0.70
d	1	0	2	234	9	0	14	260	0.9
e	30	6	16	17	190	1	0	260	0.73
f	0	0	0	0	0	260	0	260	1
g	0	0	1	2	42	0	215	260	0.83
Total	79	138	281	270	499	261	292	Accuracy	
Precision	0.24	0.57	0.64	0.87	0.38	1.00	0.74	0.65	

Table A.21 Logistic Regression, Cross-validation No. 11

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	0	0	0	0	176	84	0	260	0.00
b	0	0	3	2	66	189	0	260	0.00
c	0	0	251	0	8	1	0	260	0.97
d	0	0	0	102	2	156	0	260	0.39
e	0	0	1	3	181	75	0	260	0.70
f	0	0	7	0	0	233	20	260	0.90
g	0	0	26	97	6	1	130	260	0.50
Total	0	0	288	204	439	739	150	Accuracy	
Precision	-	-	0.87	0.50	0.41	0.32	0.87	0.49	

Table A.22 CNN, Cross-validation No. 11

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	0	5	3	33	90	129	0	260	0
b	0	43	0	4	43	170	0	260	0.17
c	0	1	198	0	11	50	0	260	0.76
d	0	28	0	161	0	71	0	260	0.62
e	0	46	15	7	83	109	0	260	0.32
f	0	0	0	0	0	245	15	260	0.94
g	1	6	55	10	8	20	160	260	0.62
Total—	1	129	271	215	235	794	175	Accuracy	
Precision	0	0.33	0.73	0.75	0.35	0.31	0.91	0.49	

Table A.23 Logistic Regression, Cross-validation No. 12

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	174	80	0	6	0	0	0	260	0.67
b	90	94	7	69	0	0	0	260	0.36
c	0	32	228	0	0	0	0	260	0.88
d	3	0	0	257	0	0	0	260	0.99
e	60	118	0	82	0	0	0	260	0.00
f	0	243	0	9	0	2	6	260	0.01
g	1	72	10	13	0	0	164	260	0.63
Total	328	639	245	436	0	2	170	Accuracy	
Precision	0.53	0.15	0.93	0.59	-	1.00	0.96	0.50	

Table A.24 CNN, Cross-validation No. 12

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	212	5	0	40	3	0	0	260	0.82
b	104	58	7	40	48	0	3	260	0.22
c	0	0	253	1	6	0	0	260	0.97
d	1	0	0	259	0	0	0	260	1.00
e	54	3	13	80	109	0	1	260	0.42
f	0	5	11	1	24	217	2	260	0.83
g	15	3	10	4	1	0	227	260	0.87
Total	386	74	294	425	191	217	233	Accuracy	
Precision	0.55	0.78	0.86	0.61	0.57	1	0.97	0.73	

Table A.25 Logistic Regression, Cross-validation No. 13

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	0	47	75	72	0	0	66	260	0.00
b	0	12	62	90	0	3	93	260	0.05
c	0	51	6	11	0	0	192	260	0.02
d	0	0	2	160	0	0	98	260	0.62
e	0	6	4	44	0	24	182	260	0.00
f	0	0	1	63	0	152	44	260	0.58
g	0	0	0	2	0	1	257	260	0.99
Total	0	116	150	442	0	180	932	Accuracy	
Precision	-	0.10	0.04	0.36	-	0.84	0.28	0.32	

Table A.26 CNN, Cross-validation No. 13

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	145	58	0	2	55	0	0	260	0.56
b	47	170	36	0	6	0	1	260	0.65
c	83	62	37	6	32	0	40	260	0.14
d	32	17	0	83	72	0	56	260	0.32
e	29	70	15	19	92	16	19	260	0.35
f	0	36	12	2	68	100	42	260	0.38
g	0	8	0	3	0	2	247	260	0.95
Total	336	421	100	115	325	118	405	Accuracy	
Precision	0.43	0.40	0.37	0.72	0.28	0.85	0.61	0.48	

Table A.27 Logistic Regression, Cross-validation No. 14

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	0	0	1	258	0	0	1	260	0.00
b	0	0	3	191	0	66	0	260	0.00
c	0	0	254	5	0	1	0	260	0.98
d	0	0	0	260	0	0	0	260	1.00
e	0	0	10	250	0	0	0	260	0.00
f	0	0	0	0	0	260	0	260	1.00
g	0	0	0	0	0	0	260	260	1.00
Total	0	0	268	964	0	327	261	Accuracy	
Precision	-	-	0.95	0.27	-	0.80	1.00	0.57	

Table A.28 CNN, Cross-validation No. 14

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	122	75	5	2	4	52	0	260	0.47
b	34	93	8	17	37	71	0	260	0.36
c	0	2	243	0	10	5	0	260	0.93
d	0	4	0	122	69	60	5	260	0.47
e	20	105	10	4	89	32	0	260	0.34
f	0	0	0	0	0	260	0	260	1
g	0	0	0	0	0	0	260	260	1
Total	176	279	266	145	209	480	265	Accuracy	
Precision	0.69	0.33	0.91	0.84	0.43	0.54	0.98	0.65	

Table A.29 Logistic Regression, Cross-validation No. 15

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	2	258	0	0	0	0	0	260	0.01
b	0	260	0	0	0	0	0	260	1.00
c	0	111	148	0	0	0	1	260	0.57
d	0	138	0	122	0	0	0	260	0.47
e	0	260	0	0	0	0	0	260	0.00
f	0	143	0	0	0	117	0	260	0.45
g	0	0	0	0	0	0	260	260	1.00
Total	2	1170	148	122	0	117	261	Accuracy	
Precision	1.00	0.22	1.00	1.00	-	1.00	1.00	0.50	

Table A.30 CNN, Cross-validation No. 15

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	64	169	0	2	0	0	25	260	0.25
b	49	101	1	0	1	62	46	260	0.39
c	19	48	63	0	0	0	130	260	0.24
d	0	64	0	188	0	0	8	260	0.72
e	15	137	0	43	7	0	58	260	0.03
f	0	0	0	0	1	259	0	260	1.00
g	0	0	0	4	0	0	256	260	0.98
Total	147	519	64	237	9	321	523	Accuracy	
Precision	0.44	0.19	0.98	0.79	0.78	0.81	0.49	0.52	

Table A.31 Logistic Regression, Cross-validation No. 16

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	192	0	64	4	0	0	0	260	0.74
b	61	10	95	82	0	4	8	260	0.04
c	0	0	260	0	0	0	0	260	1.00
d	59	0	6	195	0	0	0	260	0.75
e	165	0	11	78	0	0	6	260	0.00
f	0	0	6	2	0	228	24	260	0.88
g	2	0	8	3	0	0	247	260	0.95
Total	479	10	450	364	0	232	285	Accuracy	
Precision	0.40	1.00	0.58	0.54	-	0.98	0.87	0.62	

Table A.32 CNN, Cross-validation No. 16

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	87	57	9	0	107	0	0	260	0.33
b	27	174	2	4	53	0	0	260	0.67
c	33	42	182	0	3	0	0	260	0.70
d	64	10	1	179	6	0	0	260	0.69
e	136	19	2	0	103	0	0	260	0.40
f	2	2	0	9	0	247	0	260	0.95
g	0	0	2	0	45	0	213	260	0.82
Total	349	304	198	192	317	247	213	Accuracy	
Precision	0.25	0.57	0.92	0.93	0.32	1	1	0.65	

Table A.33 Logistic Regression, Cross-validation No. 17

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	0	195	42	1	0	3	19	260	0.00
b	0	214	40	1	0	5	0	260	0.82
c	0	107	153	0	0	0	0	260	0.59
d	0	142	0	94	0	10	0	246	0.38
e	0	188	15	0	0	44	13	260	0.00
f	0	31	19	10	0	200	0	260	0.77
g	0	188	41	5	0	1	25	260	0.10
Total	0	1065	310	111	0	263	57	Accuracy	
Precision	-	0.20	0.49	0.85	-	0.76	0.44	0.38	

Table A.34 CNN, Cross-validation No. 17

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	21	10	0	0	209	3	17	260	0.08
b	19	138	7	4	91	1	0	260	0.53
c	15	186	36	0	23	0	0	260	0.14
d	4	35	0	122	85	0	0	246	0.50
e	24	34	1	0	164	11	26	260	0.63
f	6	33	8	62	54	96	1	260	0.37
g	0	18	28	1	142	1	70	260	0.27
Total	89	454	80	189	768	112	114	Accuracy	
Precision	0.24	0.30	0.45	0.65	0.21	0.86	0.61	0.36	

Table A.35 Logistic Regression, Cross-validation No. 18

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	133	31	13	1	73	0	9	260	0.51
b	182	3	15	0	60	0	0	260	0.01
c	5	18	217	0	15	5	0	260	0.83
d	1	9	1	106	141	1	1	260	0.41
e	0	2	57	0	11	190	0	260	0.04
f	0	48	35	0	14	163	0	260	0.63
g	2	0	8	0	57	0	193	260	0.74
Total	323	111	346	107	371	359	203	Accuracy	
Precision	0.41	0.03	0.63	0.99	0.03	0.45	0.95	0.45	

Table A.36 CNN, Cross-validation No. 18

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	147	64	9	1	39	0	0	260	0.57
b	119	44	11	1	0	0	85	260	0.17
c	4	16	125	0	2	0	113	260	0.48
d	12	13	17	128	38	9	43	260	0.49
e	0	0	33	2	5	220	0	260	0.02
f	0	46	74	1	3	124	12	260	0.48
g	0	0	4	1	0	0	255	260	0.98
Total	282	183	273	134	87	353	508	Accuracy	
Precision	0.52	0.24	0.46	0.96	0.06	0.35	0.50	0.45	

Table A.37 Logistic Regression, Cross-validation No. 19

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	255	0	1	2	0	0	2	260	0.98
b	200	11	33	0	3	0	13	260	0.04
c	23	0	237	0	0	0	0	260	0.91
d	9	0	0	250	1	0	0	260	0.96
e	256	0	3	1	0	0	0	260	0.00
f	4	0	18	13	34	191	0	260	0.73
g	29	0	25	131	0	1	74	260	0.28
Total	776	11	317	397	38	192	89	Accuracy	
Precision	0.33	1.00	0.75	0.63	0.00	0.99	0.83	0.56	

Table A.38 CNN, Cross-validation No. 19

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	237	9	3	10	0	0	1	260	0.91
b	43	78	16	5	67	7	44	260	0.3
c	1	0	258	0	1	0	0	260	0.99
d	0	0	8	159	29	0	64	260	0.61
e	91	33	85	0	51	0	0	260	0.20
f	0	2	1	5	5	246	1	260	0.95
g	3	1	32	56	0	8	160	260	0.62
Total	375	123	403	235	153	261	270	Accuracy	
Precision	0.63	0.63	0.64	0.68	0.33	0.94	0.59	0.65	

Table A.39 Logistic Regression, Cross-validation No. 20

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	117	60	82	0	0	0	1	260	0.45
b	150	78	31	0	0	1	0	260	0.30
c	1	0	259	0	0	0	0	260	1.00
d	52	2	0	206	0	0	0	260	0.79
e	92	110	52	5	0	0	1	260	0.00
f	0	0	0	0	0	260	0	260	1.00
g	0	0	19	0	0	0	241	260	0.93
Total	412	250	443	211	0	261	243	Accuracy	
Precision	0.28	0.31	0.58	0.98	-	1.00	0.99	0.64	

Table A.40 CNN, Cross-validation No. 20

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	39	129	91	0	0	0	1	260	0.15
b	67	186	2	0	2	2	1	260	0.72
c	5	62	129	0	0	0	64	260	0.50
d	173	26	0	38	18	0	5	260	0.15
e	79	142	17	4	18	0	0	260	0.07
f	0	0	0	0	1	259	0	260	1.00
g	0	26	0	0	0	0	234	260	0.9
Total	363	571	239	42	39	261	305	Accuracy	
Precision	0.11	0.33	0.54	0.90	0.46	0.99	0.77	0.50	

Table A.41 Logistic Regression, Cross-validation No. 21

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	251	0	4	1	0	0	4	260	0.97
b	112	0	88	25	0	5	30	260	0.00
c	5	0	255	0	0	0	0	260	0.98
d	60	0	0	194	0	0	6	260	0.75
e	129	0	43	4	0	80	4	260	0.00
f	0	0	2	0	0	257	1	260	0.99
g	1	0	107	3	0	3	146	260	0.56
Total	558	0	499	227	0	345	191	Accuracy	
Precision	0.45	-	0.51	0.85	-	0.74	0.76	0.61	

Table A.42 CNN, Cross-validation No. 21

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	259	0	0	1	0	0	0	260	1.00
b	153	47	24	32	4	0	0	260	0.18
c	5	0	249	1	3	0	2	260	0.96
d	51	0	0	193	0	0	16	260	0.74
e	22	33	7	13	133	48	4	260	0.51
f	0	7	0	2	16	235	0	260	0.90
g	111	2	0	3	0	0	144	260	0.55
Total	601	89	280	245	156	283	166	Accuracy	
Precision	0.43	0.53	0.89	0.79	0.85	0.83	0.87	0.69	

Table A.43 Logistic Regression, Cross-validation No. 22

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	0	2	0	26	232	0	0	260	0.00
b	0	109	0	47	104	0	0	260	0.42
c	0	17	5	11	218	9	0	260	0.02
d	0	0	0	165	86	9	0	260	0.63
e	0	30	1	0	208	21	0	260	0.80
f	0	85	0	74	55	35	11	260	0.13
g	0	6	57	5	149	0	43	260	0.17
Total	0	249	63	328	1052	74	54	Accuracy	
Precision	-	0.44	0.08	0.50	0.20	0.47	0.80	0.31	

Table A.44 CNN, Cross-validation No. 22

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	138	82	0	7	33	0	0	260	0.53
b	14	184	0	7	13	41	1	260	0.71
c	49	30	69	19	73	19	1	260	0.27
d	17	40	1	144	20	38	0	260	0.55
e	3	76	1	7	61	110	2	260	0.23
f	0	67	1	0	1	184	7	260	0.71
g	3	27	10	7	6	10	197	260	0.76
Total	224	506	82	191	207	402	208	Accuracy	
Precision	0.62	0.36	0.84	0.75	0.29	0.46	0.95	0.54	

Table A.45 Logistic Regression, Cross-validation No. 23

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	260	0	0	0	0	0	0	260	1.00
b	232	0	4	1	3	20	0	260	0.00
c	77	0	183	0	0	0	0	260	0.70
d	59	0	0	201	0	0	0	260	0.77
e	260	0	0	0	0	0	0	260	0.00
f	9	0	1	0	0	250	0	260	0.96
g	179	0	1	4	0	0	76	260	0.29
Total	1076	0	189	206	3	270	76	Accuracy	
Precision	0.24	-	0.97	0.98	0.00	0.93	1.00	0.53	

Table A.46 CNN, Cross-validation No. 23

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	65	0	0	28	167	0	0	260	0.25
b	3	70	23	5	132	27	0	260	0.27
c	0	0	256	0	1	0	3	260	0.98
d	0	1	0	258	1	0	0	260	0.99
e	1	0	65	2	192	0	0	260	0.74
f	0	4	1	1	1	253	0	260	0.97
g	0	1	0	4	18	15	222	260	0.85
Total	69	76	345	298	512	295	225	Accuracy	
Precision	0.94	0.92	0.74	0.87	0.38	0.86	0.99	0.72	

Table A.47 Logistic Regression, Cross-validation No. 24

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	0	56	0	0	204	0	0	260	0.00
b	0	19	0	0	241	0	0	260	0.07
c	0	30	127	0	103	0	0	260	0.49
d	0	0	0	17	243	0	0	260	0.07
e	0	2	0	0	258	0	0	260	0.99
f	0	3	0	0	106	151	0	260	0.58
g	0	23	0	0	0	0	237	260	0.91
Total	0	133	127	17	1155	151	237	Accuracy	
Precision	-	0.14	1.00	1.00	0.22	1.00	1.00	0.44	

Table A.48 CNN, Cross-validation No. 24

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	211	16	0	4	29	0	0	260	0.81
b128	79	1	6	45	0	1	260	0.30	
c	69	9	158	0	20	0	4	260	0.61
d	2	2	0	233	23	0	0	260	0.90
e	175	2	0	20	62	0	1	260	0.24
f	0	1	0	0	3	256	0	260	0.98
g	0	0	0	0	0	0	260	260	1
Total	585	109	159	263	182	256	266	Accuracy	
Precision	0.36	0.72	0.99	0.89	0.34	1	0.98	0.69	

Table A.49 Logistic Regression, Cross-validation No. 25

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	260	0	0	0	0	0	0	260	1.00
b	260	0	0	0	0	0	0	260	0.00
c	260	0	0	0	0	0	0	260	0.00
d	260	0	0	0	0	0	0	260	0.00
e	201	0	0	0	0	0	0	201	0.00
f	260	0	0	0	0	0	0	260	0.00
g	260	0	0	0	0	0	0	260	0.00
Total	1761	0	0	0	0	0	0	Accuracy	
Precision	0.15	-	-	-	-	-	-	0.15	

Table A.50 CNN, Cross-validation No. 25

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	69	82	53	4	2	14	36	260	0.27
b	48	122	59	16	0	15	0	260	0.47
c	1	0	122	0	28	94	15	260	0.47
d	3	34	0	187	3	6	27	260	0.72
e	55	75	42	3	22	3	1	201	0.11
f	0	4	0	26	11	218	1	260	0.84
g	0	4	0	7	0	0	249	260	0.96
Total	176	321	276	243	66	350	329	Accuracy	
Precision	0.39	0.38	0.44	0.77	0.33	0.62	0.76	0.56	

Table A.51 Logistic Regression, Cross-validation No. 26

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	0	0	149	6	105	0	0	260	0.00
b	0	0	186	24	45	0	5	260	0.00
c	0	0	260	0	0	0	0	260	1.00
d	0	0	6	202	51	0	1	260	0.78
e	0	0	233	0	27	0	0	260	0.10
f	0	0	246	0	14	0	0	260	0.00
g	0	0	62	0	1	0	197	260	0.76
Total	0	0	1142	232	243	0	203	Accuracy	
Precision	-	-	0.23	0.87	0.11	-	0.97	0.38	

Table A.52 CNN, Cross-validation No. 26

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	85	25	59	17	74	0	0	260	0.33
b	1	87	80	3	89	0	0	260	0.33
c	0	0	260	0	0	0	0	260	1
d	4	31	2	138	84	1	0	260	0.53
e	43	21	60	53	83	0	0	260	0.32
f	0	2	16	1	69	172	0	260	0.66
g	9	30	3	0	3	0	215	260	0.83
Total	142	196	480	212	402	173	215	Accuracy	
Precision	0.60	0.44	0.54	0.65	0.21	0.99	1	0.57	

Table A.53 Logistic Regression, Cross-validation No. 27

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	253	0	0	7	0	0	0	260	0.97
b	34	0	184	37	0	2	3	260	0.00
c	0	0	260	0	0	0	0	260	1.00
d	0	0	0	260	0	0	0	260	1.00
e	11	0	249	0	0	0	0	260	0.00
f	0	0	65	25	0	170	0	260	0.65
g	0	0	0	0	0	0	260	260	1.00
Total	298	0	758	329	0	172	263	Accuracy	
Precision	0.85	-	0.34	0.79	-	0.99	0.99	0.66	

Table A.54 CNN, Cross-validation No. 27

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	220	1	0	1	38	0	0	260	0.85
b	92	29	2	0	81	0	56	260	0.11
c	0	40	215	0	0	0	5	260	0.83
d	0	0	0	260	0	0	0	260	1
e	0	0	0	0	245	0	15	260	0.94
f	0	28	3	9	16	165	39	260	0.63
g	0	0	0	0	0	0	260	260	1
Total	312	98	220	270	380	165	375	Accuracy	
Precision	0.71	0.30	0.98	0.96	0.64	1	0.69	0.77	

Table A.55 Logistic Regression, Cross-validation No. 28

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	215	43	2	0	0	0	0	260	0.83
b	23	228	0	9	0	0	0	260	0.88
c	0	95	165	0	0	0	0	260	0.63
d	0	0	0	260	0	0	0	260	1.00
e	8	251	0	0	1	0	0	260	0.00
f	0	98	0	0	0	162	0	260	0.62
g	0	0	0	0	0	0	260	260	1.00
Total	246	715	167	269	1	162	260	Accuracy	
Precision	0.87	0.32	0.99	0.97	1.00	1.00	1.00	0.71	

Table A.56 CNN, Cross-validation No. 28

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	137	101	0	3	19	0	0	260	0.53
b	0	254	0	1	0	0	5	260	0.98
c	0	77	183	0	0	0	0	260	0.70
d	0	3	0	257	0	0	0	260	0.99
e	1	43	0	0	215	0	1	260	0.83
f	0	63	0	0	0	197	0	260	0.76
g	0	0	0	0	0	0	260	260	1
Total	138	541	183	261	234	197	266	Accuracy	
Precision	0.99	0.47	1	0.98	0.92	1	0.98	0.83	

Table A.57 Logistic Regression, Cross-validation No. 29

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	0	236	0	24	0	0	0	260	0.00
b	0	257	0	3	0	0	0	260	0.99
c	0	2	258	0	0	0	0	260	0.99
d	0	0	0	260	0	0	0	260	1.00
e	0	26	5	51	9	0	169	260	0.03
f	0	14	0	0	0	246	0	260	0.95
g	0	0	0	0	0	0	260	260	1.00
Total	0	535	263	338	9	246	429	Accuracy	
Precision	-	0.48	0.98	0.77	1.00	1.00	0.61	0.71	

Table A.58 CNN, Cross-validation No. 29

Actual	Predicted posture								
posture	a	b	c	d	e	f	g	Total	Recall
a	50	206	0	0	4	0	0	260	0.19
b	0	195	4	0	61	0	0	260	0.75
c	2	0	247	0	11	0	0	260	0.95
d	0	0	0	249	0	0	11	260	0.96
e	23	17	38	3	54	0	125	260	0.21
f	0	72	0	0	166	22	0	260	0.08
g	0	3	0	1	0	0	256	260	0.98
Total	75	493	289	253	296	22	392	Accuracy	
Precision	0.67	0.40	0.85	0.98	0.18	1	0.65	0.59	

Appendix B

Python Code for Chapter III

B.1 Logistic Regression

```
class LogisticReg(Model):
    def __init__(self, shape_size, class_size):
        self.x = tf.placeholder(tf.float32, (None, shape_size, shape_size, 1))
        self.y = tf.placeholder(tf.int32, (None))
        super(LogisticReg, self).__init__(shape_size, class_size)
    def get_hypothesis(self, x, shape_size=400, class_size=10):
        feature_size = shape_size * shape_size
        xr = tf.reshape(self.x, [-1, feature_size])
        W = tf.Variable(tf.zeros([feature_size, class_size]))
        b = tf.Variable(tf.zeros([class_size]))
        return tf.matmul(xr, W) + b
    def define_training_operation(self):
        cost = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logit
        ↪   s_v2(labels=self.one_hot_y, logits=self.logits))
        self.training_operation = tf.train.GradientDescentOptimizer(0.01).minimize(cost)
        ↪   # learning_rate = 0.01
```

B.2 Modified Lenet-5

```
class LeNet(Model):
    def __init__(self, shape_size, class_size, features=100):
        self.x = tf.placeholder(tf.float32, (None, features, features, 1))
        self.y = tf.placeholder(tf.int32, (None))
        super(LeNet, self).__init__(shape_size, class_size)
    def get_shape_str(self, shape):
        shape_list = shape.as_list()
        for i in range(0, len(shape_list)):
            if shape_list[i] == None:
                shape_list[i] = "?"
            else:
                shape_list[i] = str(shape_list[i])
        result = ', '.join(map(str, shape_list))
        return "("+ result + ")"
    def print_shape_msg(self, msg, input_shape, output_shape):
        if input_shape is not None:
            msg += " : Input : " + self.get_shape_str(input_shape)
        if output_shape is not None:
            msg += " Output : " + self.get_shape_str(output_shape)
        print (msg)
    def define_training_operation(self):
        self.cross_entropy = tf.nn.softmax_cross_entropy_with_logits(logits=self.
        ↪  logits, labels=self.one_hot_y)
        self.loss_operation = tf.reduce_mean(self.cross_entropy)
        self.optimizer = tf.train.AdamOptimizer(learning_rate = 0.001)
        self.training_operation = self.optimizer.minimize(self.loss_operation)
    def get_hypothesis(self, x, shape_size=400, class_size=10):
        # Hyperparameters
        mu = 0
        sigma = 0.1
        layer_depth = {
            'layer_1' : 6,
            'layer_2' : 16,
            'layer_3' : 120,
            'layer_f1' : 84
        }
```

```
# Layer 1: Convolutional. Input = 32x32x1. Output = 28x28x6.
conv1_w = tf.Variable(tf.truncated_normal(shape = [5,5,1,6],mean = mu, stddev =
 ↪  sigma))
conv1_b = tf.Variable(tf.zeros(6))
conv1 = tf.nn.conv2d(x,conv1_w, strides = [1,1,1,1], padding = 'VALID') +
 ↪  conv1_b
# Activation.
conv1 = tf.nn.relu(conv1)
# Pooling. Input = 28x28x6. Output = 14x14x6.
pool_1 = tf.nn.max_pool(conv1,ksize = [1,2,2,1], strides = [1,2,2,1], padding =
 ↪  'VALID')
# Layer 2: Convolutional. Output = 10x10x16.
conv2_w = tf.Variable(tf.truncated_normal(shape = [5,5,6,16], mean = mu, stddev
 ↪  = sigma))
conv2_b = tf.Variable(tf.zeros(16))
conv2 = tf.nn.conv2d(pool_1, conv2_w, strides = [1,1,1,1], padding = 'VALID') +
 ↪  conv2_b
# Activation.
conv2 = tf.nn.relu(conv2)
# Pooling. Input = 10x10x16. Output = 5x5x16.
pool_2 = tf.nn.max_pool(conv2, ksize = [1,2,2,1], strides = [1,2,2,1], padding =
 ↪  'VALID')
# Flatten. Input = 5x5x16. Output = 400.
fc1 = flatten(pool_2)
# Layer 3: Fully Connected. Input = 400. Output = 120.
# fc1_w = tf.Variable(tf.truncated_normal(shape = (400,120), mean = mu, stddev =
 ↪  sigma))
fc1_w = tf.Variable(tf.truncated_normal(shape = (shape_size,120), mean = mu,
 ↪  stddev = sigma))
fc1_b = tf.Variable(tf.zeros(120))
fc1_fully = tf.matmul(fc1,fc1_w) + fc1_b
# Activation.
fc1_act = tf.nn.relu(fc1_fully)
# Layer 4: Fully Connected. Input = 120. Output = 84.
fc2_w = tf.Variable(tf.truncated_normal(shape = (120,84), mean = mu, stddev =
 ↪  sigma))
fc2_b = tf.Variable(tf.zeros(84))
fc2 = tf.matmul(fc1_act,fc2_w) + fc2_b
# Activation.
```

```
        fc2_act = tf.nn.relu(fc2)
        # Layer 5: Fully Connected. Input = 84. Output = 10.
        # fc3_w = tf.Variable(tf.truncated_normal(shape = (84, 10), mean = mu , stddev =
         ↪  sigma))
        fc3_w = tf.Variable(tf.truncated_normal(shape = (84, class_size), mean = mu ,
         ↪  stddev = sigma))
        # fc3_b = tf.Variable(tf.zeros(10))
        fc3_b = tf.Variable(tf.zeros(class_size))
        logits = tf.matmul(fc2_act, fc3_w) + fc3_b
        return logits

    def save_graph(self):
        with tf.Session() as sess:
            logis = self.get_hypothesis(self.x, 7744, 7)
            logs_path = '/tmp/deep/example'
            summary_writer = tf.summary.FileWriter(logs_path,
             ↪  graph=tf.get_default_graph())
```

Appendix C

Python Code for Chapter IV

C.1 Data Pre-processing

```
import json
import random
import h5py
import numpy as np
def video_to_array(video_path,
                   resize=None,
                   start_frame=0,
                   end_frame=None,
                   length=None,
                   dim_ordering='th'):
    ''' Convert the video at the path given in to an array
    Args:
        video_path (string): path where the video is stored
        resize (Optional[tupple(int)]): desired size for the output video.
            Dimensions are: height, width
        start_frame (Optional[int]): Number of the frame to start to read
            the video
        end_frame (Optional[int]): Number of the frame to end reading the
            video.
        length (Optional[int]): Number of frames of length you want to read
```

```
        the video from the start_frame. This override the end_frame
        given before.
Returns:
    video (nparray): Array with all the data corresponding to the video
                     given. Order of dimensions are: channels, length
                     (temporal), height, width.
Raises:
    Exception: If the video could not be opened
'''
import cv2
if cv2.__version__ >= '3.0.0':
    CAP_PROP_FRAME_COUNT = cv2.CAP_PROP_FRAME_COUNT
    CAP_PROP_POS_FRAMES = cv2.CAP_PROP_POS_FRAMES
else:
    CAP_PROP_FRAME_COUNT = cv2.cv.CV_CAP_PROP_FRAME_COUNT
    CAP_PROP_POS_FRAMES = cv2.cv.CV_CAP_PROP_POS_FRAMES
if dim_ordering not in ('th', 'tf'):
    raise Exception('Invalid dim_ordering')
cap = cv2.VideoCapture(video_path)
if not cap.isOpened():
    raise Exception('Could not open the video')
num_frames = int(cap.get(CAP_PROP_FRAME_COUNT))
if start_frame >= num_frames or start_frame < 0:
    raise Exception('Invalid initial frame given')
# Set up the initial frame to start reading
cap.set(CAP_PROP_POS_FRAMES, start_frame)
# Set up until which frame to read
if end_frame:
    end_frame = end_frame if end_frame < num_frames else num_frames
elif length:
    end_frame = start_frame + length
    end_frame = end_frame if end_frame < num_frames else num_frames
else:
    end_frame = num_frames
if end_frame < start_frame:
    raise Exception('Invalid ending position')
frames = []
for i in range(start_frame, end_frame):
    ret, frame = cap.read()
```

```
        if cv2.waitKey(1) & 0xFF == ord('q') or not ret:
            return None
        if resize:
            # The resize of CV2 requires pass firts width and then height
            frame = cv2.resize(frame, (resize[1], resize[0]))
        frames.append(frame)
    video = np.array(frames, dtype=np.float32)
    return video
def save_image(filename, image_array):
    import scipy.misc
    scipy.misc.imsave(filename, image_array)
def get_json_dict(filename):
    with open(filename) as data_file:
        return json.load(data_file)
    return None
class data(object):
    data_name = None
    config = None
    train_years = []
    test_years = ['2016']
    hd5py_handle = {}
    def __init__(self):
        print "hello : %s" %(self.data_name)
        self.config = get_json_dict('config.json')
        feature_path = self.config['feature_path']
        years = []
        years += self.test_years[:]
        years += self.train_years[:]
        for y in years:
            self.hd5py_handle[y] = h5py.File(feature_path + y + ".h5", "r")
    def __del__(self):
        print "goodbye : %s" %(self.data_name)
        for key in self.hd5py_handle.keys():
            self.hd5py_handle[key].close()
    def get_frame_data(self, filename, video_name, size):
        h5py_data = self.hd5py_handle[filename]
        offset = 0
        batch_x = []
        max_value = len(h5py_data[video_name]) - size
```

```
        offset = random.randint(0, max_value)
        batch_x.append(h5py_data[video_name][offset:offset+size])
        return batch_x
    def get_batch_data(self, data_type, data_name, n_input, sheets=None):
        batch_x = None
        batch_y = None
        names = self.get_video_names(data_type, data_name)
        count = 0
        for video_name in names:
            if sheets is not None:
                sheets['file'].append(video_name)
            tmp_x = self.get_frame_data(data_name, video_name, n_input)
            tmp_y = self.get_label_data(data_type, video_name, len(tmp_x))
            if count == 0:
                batch_x = tmp_x
                batch_y = tmp_y
            else:
                batch_x = np.concatenate((batch_x, tmp_x), axis=0)
                batch_y = np.concatenate((batch_y, tmp_y), axis=0)
            count += 1
        return batch_x, batch_y
    def get_label_file(self, type_str):
        return self.config['label_path'] + type_str + "-" + self.data_name + ".h5"
    def get_label_data(self, type_str, video_name, count):
        data = None
        f = h5py.File(self.get_label_file(type_str))
        if video_name in f.keys():
            data = f[video_name][0]
        f.close()
        result = []
        for i in range(0, count):
            result.append(data)
        return result
    def get_video_names(self, type_str, year=None):
        names = [];
        f = h5py.File(self.get_label_file(type_str))
        if year == None:
            names = f.keys()
        else:
```

```
            for k in f.keys():
                if k.find(year+'_') != -1:
                    names.append(k)
        f.close()
        return names
    def get_class_count(self):
        return self.config[self.data_name + '_class_count']
    def get_data_name(self):
        return self.data_name
    def get_labels(self):
        return range(0, self.get_class_count())
    def get_data_sets(self, data_type, years, n_input, sheets=None):
        x = None
        y = None
        for l in years:
            batch_x, batch_y = self.get_batch_data(data_type, l, n_input, sheets)
            if x is None:
                x = batch_x
                y = batch_y
            else:
                x = np.concatenate((x, batch_x), axis=0)
                y = np.concatenate((y, batch_y), axis=0)
        return x, y
    def get_train_data(self, n_input, sheets=None):
        return self.get_data_sets('train', self.train_years, n_input, sheets)
    def get_test_data(self, n_input, sheets=None):
        return self.get_data_sets('test', self.test_years, n_input, sheets)
class owas(data):
    data_name = "owas"
    train_years = ['2007', '2009', '2012', '2013', '2014']
    def __init__(self):
        super(owas, self).__init__()
class rula(data):
    data_name = "rula"
    train_years = ['2007', '2009', '2012', '2013', '2014']
    def __init__(self):
        super(rula, self).__init__()
class reba(data):
    data_name = "reba"
```

```
    train_years = ['2013', '2014']
    def __init__(self):
        super(reba, self).__init__()
class mnist(object):
    mnist = None;
    def __init__(self, mnist):
        self.mnist = mnist
    def get_class_count(self):
        return 10
    def get_data_name(self):
        return 'mnist'
    def get_labels(self):
        return range(0, 10)
    def get_train_data(self, n_input):
        batch_size = 100
        batch_x, batch_y = self.mnist.train.next_batch(batch_size)
        batch_x = batch_x.reshape(len(batch_x), 28, 28)
        return batch_x, batch_y
    def get_test_data(self, n_input):
        test_x = self.mnist.test.images
        test_y = self.mnist.test.labels
        test_x = test_x.reshape(len(test_x), 28, 28)
        return test_x, test_y
```

C.2 Model(RNN, LSTM)

```
#-*- coding: utf-8 -*-
import sys
import json
import matplotlib
# Force matplotlib to not use any Xwindows backend.
matplotlib.use('Agg')
import h5py
import warnings
import numpy as np
import tensorflow as tf
from tensorflow.contrib import rnn
```

```
import random
import time
import data
import itertools
import matplotlib.pyplot as plt
from sklearn.utils import shuffle
from sklearn.metrics import precision_recall_fscore_support as score
from sklearn.metrics import confusion_matrix
from tensorflow.contrib import rnn
def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):
    """
    This function prints and plots the confusion matrix.
    Normalization can be applied by setting `normalize=True`.
    """
    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)
    fmt = '.2f' if normalize else 'd'
    thresh = cm.max() / 2.
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, format(cm[i, j], fmt),
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")
    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')

def RNN(x, weights, biases, n_input, n_hidden):
    x = tf.unstack(x, n_input, 1)
    #x = tf.unstack(x, 10, 1)
    #x = tf.unstack(x, 28, 1) #<<== MNIST
```

```
    """
    RNN cell composed sequentially of multiple simple cells.
    """
    # 2-layer LSTM, each layer has n_hidden units.
    rnn_cell = rnn.MultiRNNCell([rnn.BasicLSTMCell(n_hidden),rnn.
     ↪  BasicLSTMCell(n_hidden)])
    # 1-layer LSTM with n_hidden units but with lower accuracy.
    # Uncomment line below to test but comment out the 2-layer rnn.MultiRNNCell above
    # rnn_cell = rnn.BasicLSTMCell(n_hidden)
    """
    A pair (outputs, state) where:
     - outputs is a length T list of outputs (one for each input), or a nested tuple of
↪   such elements.
     - state is the final state

    https://www.tensorflow.org/api_docs/python/tf/nn/static_rnn
    """
    # generate prediction
    outputs, states = rnn.static_rnn(rnn_cell, x, dtype=tf.float32)
    # there are n_input outputs but
    # we only want the last output
    return tf.matmul(outputs[-1], weights['out']) + biases['out']
class LSTM(object):
    dataobj = None
    n_input = None
    def __init__(self, dataobj, n_input=10, n_features=4096, n_iters=50000):
        self.dataobj = dataobj
        class_size = self.dataobj.get_class_count()
        learning_rate = 0.0001
        n_hidden = 512
        self.n_input = n_input
        self.n_iters = n_iters
        self.n_display_step = 100
        self.n_batch_step = 1000
        # tf Graph input
        self.x = tf.placeholder("float", [None, n_input, n_features])
        self.y = tf.placeholder("float", [None, class_size])
        # RNN output node weights and biases
        self.weights = {
```

```
            'out': tf.Variable(tf.random_normal([n_hidden, class_size]))
        }
        self.biases = {
            'out': tf.Variable(tf.random_normal([class_size]))
        }
        self.pred = RNN(self.x, self.weights, self.biases, n_input, n_hidden)
        # Loss and optimizer
        self.cost = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits
        ↪   (logits=self.pred, labels=self.y))
        self.optimizer = tf.train.AdamOptimizer(learning_rate=learning_rate)
        ↪   .minimize(self.cost)
        #optimizer = tf.train.GradientDescentOptimizer(learning_rate
        ↪   =learning_rate).minimize(cost)
        #optimizer = tf.train.RMSPropOptimizer(learning_rate=learning_rate)
        ↪   .minimize(cost)
        # Create a summary to monitor cost tensor
        tf.summary.scalar("loss", self.cost)
        # Merge all summaries into a single op
        self.merged_summary_op = tf.summary.merge_all()
        self.saver = tf.train.Saver()
    def get_input_size(self):
        return self.n_input
    def get_checkpoint_path(self):
        return "checkpoints/" + self.dataobj.get_data_name()
    def predict(self, X_data):
        num_examples = len(X_data)
        sess = tf.get_default_session()
        return sess.run(tf.argmax(self.pred,1), feed_dict={self.x:X_data})
    def get_metrics(self, answer_y, predicted_y, labels):
        with warnings.catch_warnings():
            warnings.simplefilter("ignore")
            precision, recall, fscore, _ = score(answer_y, predicted_y, labels=labels)
            return precision, recall, fscore
    def evaluate(self, X_data, y_data):
        total_accuracy = 0
        predicted_y = self.predict(X_data)
        answer_y = np.argmax(y_data, axis=1)
        labels = self.dataobj.get_labels()
        precision, recall, fscore = self.get_metrics(answer_y, predicted_y, labels)
```

```
        for l in range(len(labels)):
            print "[%s] : precision : %f, recall : %f, fscore : %f" % (str(labels[l]),
             ↪  precision[l], recall[l], fscore[l])
        return float(np.sum(predicted_y == answer_y)) / float(len(X_data))
    """ training """
    def train(self, logs_path=None):
        # Initializing the variables
        init = tf.global_variables_initializer()
        n_input = self.get_input_size()
        batch_x = None
        batch_y = None
        summary_writer = None
        # Launch the graph
        with tf.Session() as sess:
            sess.run(init)
            if logs_path is not None:
                # op to write logs to Tensorboard
                summary_writer = tf.summary.FileWriter(logs_path,
                 ↪  graph=tf.get_default_graph())
            for step in range(0, self.n_iters+1):
                """ n_batch_step data read """
                if step % self.n_batch_step == 0 and step != self.n_iters:
                    batch_x, batch_y = self.dataobj.get_train_data(n_input)
#                 batch_x, batch_y = self.dataobj.get_train_data(n_input)
                sess.run(self.optimizer, feed_dict={self.x: batch_x, self.y: batch_y})
                loss = sess.run(self.cost, feed_dict={self.x: batch_x, self.y: batch_y})
                if summary_writer is not None:
                    # Write logs at every iteration
                    summary = sess.run(self.merged_summary_op, feed_dict={self.x:
                     ↪  batch_x, self.y: batch_y})
                    summary_writer.add_summary(summary, step)
                print "step : %d, loss : %f" % (step, loss)
                sys.stdout.flush()
                if step !=0 and step % self.n_display_step == 0:
                    arr = self.evaluate(batch_x, batch_y)
                    print("Training Accuracy = {:.3f}".format(arr))
            checkpoint = self.get_checkpoint_path()
            self.saver.save(sess, checkpoint)
            print "Model saved : ", checkpoint
```

```
        print ("Optimization Finished!")
        if logs_path is not None:
            print ("Run the command line: tensorboard --logdir="+logs_path)
    def get_accuracy_data(self, X_test, y_test, type_str, sheets=None):
        data = {}
        labels = self.dataobj.get_labels()
        y_test = np.argmax(y_test, axis=1)
        sess = tf.get_default_session()
        y_predict = self.predict(X_test)
        if sheets is not None:
            sheets['answer'] = y_test[:]
            sheets['predicted'] = y_predict[:]
        precision, recall, fscore = self.get_metrics(y_test, y_predict, labels)
        accuracy = float(np.sum(y_predict == y_test)) / float(len(y_test))
        cnf_matrix = confusion_matrix(y_test, y_predict, labels)
        data["accuracy"] = accuracy
        data["precision"] = precision.tolist()
        data["recall"] = recall.tolist()
        data["fscore"] = fscore.tolist()
        data["confusion"] = cnf_matrix.tolist()
        # save cm graph
        plt.figure()
        plot_confusion_matrix(cnf_matrix, classes=labels, normalize=False,
            title='Confusion matrix['+type_str+'-'+str(self.dataobj
             ↪   .get_data_name())+']')
        plt.savefig('results/cm/'+type_str+'-'+self.dataobj
         ↪   .get_data_name()+'-type1.png', bbox_inches='tight');
        plt.figure()
        plot_confusion_matrix(cnf_matrix, classes=labels, normalize=True,
            title='Normalized confusion matrix['+type_str+'-'+str(self.dataobj
             ↪   .get_data_name())+']')
        plt.savefig('results/cm/'+type_str+'-'+self.dataobj
         ↪   .get_data_name()+'-type2.png', bbox_inches='tight');
        return data
    def save_test_data(self, type_str):
        x = y = None
        sheets =  {}
        sheets['file'] = []
        sheets['answer'] = []
```

```
        sheets['predicted'] = []
        if type_str == 'test':
            x, y  = self.dataobj.get_test_data(self.get_input_size(), sheets)
        else:
            x, y  = self.dataobj.get_train_data(self.get_input_size(), sheets)
        print np.shape(x)
        print np.shape(y)
        result = self.get_accuracy_data(x, y, type_str, sheets)
        """ data save """
        file_name = 'results/'+type_str+'-' + self.dataobj.get_data_name() + '.csv'
        with open(file_name, 'w') as outfile:
            for s in range(len(sheets['file'])):
                outfile.write(str(sheets['file'][s]) + ',' + str(sheets['answer'][s]) +
                 ↪  ',' + str(sheets['predicted'][s]) + '\n')
        file_name = 'results/'+type_str+'-' + self.dataobj.get_data_name() + '.json'
        with open(file_name, 'w') as outfile:
                json.dump(result, outfile)
        print json.dumps(result)
    def test(self):
        with tf.Session() as sess:
            self.saver.restore(sess, self.get_checkpoint_path())
            self.save_test_data('test')
            self.save_test_data('train')
```

Appendix D

Python Code for Chapter V

D.1 Linear Regression

```
class Reg(Model):
    def __init__(self, n_feature, modelname, learning_rate=0.01):
        self.learning_rate = learning_rate
        super(Reg, self).__init__(n_feature, modelname)
    def H(self, n_feature):
        # tf Graph Input
        self.tf_x = tf.placeholder("float")
        self.tf_y = tf.placeholder("float")

        W = tf.get_variable(initializer=tf.truncated_normal([n_feature, 1], stddev=0.1),
        ↪   name = "W")
        b = tf.get_variable(initializer=tf.constant(0.1, shape=[1]), name = "B")
        # Construct a linear model
        return tf.matmul(self.tf_x, W) + b

    def train(self, x, y):
        n_samples = tf.cast(tf.shape(self.tf_x)[0], tf.float32)

        cost = tf.reduce_sum(tf.pow(self.pred - self.tf_y, 2))/(2*n_samples)
        optimizer = tf.train.AdamOptimizer(self.learning_rate).minimize(cost)
```

```
        # Start training
        with tf.Session() as sess:
            sess.run(tf.global_variables_initializer())
            for _ in range(self.iter):
                sess.run(optimizer, feed_dict={self.tf_x:x, self.tf_y:y})
                loss = sess.run(cost, feed_dict={self.tf_x:x, self.tf_y:y})
                result = sess.run(self.pred, feed_dict={self.tf_x:x, self.tf_y:y})
                print _, loss
            self.save_checkpoint(sess)
    def feed_data(self, x, y):
        return {self.tf_x:x, self.tf_y:y}
    def predict(self, x, y, sess):
        y_predicted = sess.run(self.pred, feed_dict=self.feed_data(x, y))
        return y.flatten(), y_predicted.flatten()
```

D.2 RGLM

```
class RGLM(Reg):
    def __init__(self, n_feature, modelname):
        self.n_layerout = 5
        super(RGLM, self).__init__(n_feature, modelname, 0.05)

    def H(self, n_feature):
        # tf Graph Input
        self.tf_x = tf.placeholder("float")
        self.tf_y = tf.placeholder("float")
        W1 = tf.get_variable(initializer=tf.truncated_normal([n_feature,
         ↪  self.n_layerout], stddev=0.1), name = "W1")
        b1 = tf.get_variable(initializer=tf.constant(0.1, shape=[self.n_layerout]), name
         ↪  = "B1")
        layer1 = tf.matmul(self.tf_x, W1) + b1
        W2 = tf.get_variable(initializer=tf.truncated_normal([self.n_layerout, 1],
         ↪  stddev=0.1), name = "W2")
        b2 = tf.get_variable(initializer=tf.constant(0.1, shape=[1]), name = "B2")
        layer2 = tf.matmul(layer1, W2) + b2
        return layer2
```

D.3 RNN

```
class RNN(Model):
    def __init__(self, n_feature, modelname, n_class=2000, learning_rate=0.001):
        self.n_timestep = n_feature      # rnn time step
        self.n_input = 1           # rnn input size
        self.n_cellsize = 128       # rnn cell size
        self.learning_rate = learning_rate
        self.n_class = n_class
        super(RNN, self).__init__(n_feature, modelname)
        self.cost, self.optimizer = self.optimizer()

    def H(self, n_feature):
        # tensorflow placeholders
        self.tf_x = tf.placeholder(tf.float32, [None, self.n_timestep, self.n_input])
        ↪   # shape(batch, 5, 1)
        self.tf_y = tf.placeholder(tf.float32, [None, self.n_class])
        weight = tf.get_variable(initializer=tf.truncated_normal([self.n_cellsize,
        ↪   self.n_class], stddev=0.1), name = "W")
        bias = tf.get_variable(initializer=tf.constant(0.1, shape=[self.n_class]), name
        ↪   = "B")
        # RNN
        rnn_cell = tf.contrib.rnn.BasicRNNCell(num_units=self.n_cellsize)
        x = tf.unstack(self.tf_x, self.n_timestep, 1)
        outputs, states = tf.contrib.rnn.static_rnn(rnn_cell, x, dtype=tf.float32)
        return tf.matmul(outputs[-1], weight) + bias

    def feed_data(self, x, y):
        x = x.reshape(x.shape + (1,))
        y = self.one_hot(y, self.n_class)
        return {self.tf_x:x, self.tf_y:y}
    def optimizer(self):
        cost = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(logits=self.pred,
        ↪   labels=self.tf_y))
        optimizer = tf.train.AdamOptimizer(self.learning_rate).minimize(cost)
        return cost, optimizer
    def train(self, x, y):
        sess = tf.Session()
```

```
        sess.run(tf.global_variables_initializer())      # initialize var in graph
        feed_dict = self.feed_data(x, y)
        for step in range(self.iter):
            sess.run(self.optimizer, feed_dict)
            loss = sess.run(self.cost, feed_dict)
            print step, loss
        self.save_checkpoint(sess)

    def predict(self, x, y, sess):
        y_predicted = sess.run(tf.argmax(tf.nn.softmax(self.pred),1),
        ↪   feed_dict=self.feed_data(x, y))
        return y.astype(int).flatten(), y_predicted
```

국문초록

인간공학 연구는 관찰, 측정, 분석을 통해 이루어진다. 관찰, 측정, 분석과 관련된 기술의 발달로 인해 인간공학 연구 역시 발달해 왔다. 딥러닝 기술은 인공지능 개발을 위한 핵심기술이다. 딥러닝을 활용하여 인간의 관찰, 측정, 분석 능력을 보완하고, 대체 하려는 다양한 시도들이 이루어지고 있다. 이러한 딥러닝은 인간공학 연구 과정의 다양한 단계에 적용될 수 있다. 이에, 본 연구에서는 인간공학 연구에 딥러닝을 응용할 수 있는 방안을 마련하기 위해 다양한 시도를 하였다.

본 연구에서는 수치 데이터, 이미지 데이터, 영상 데이터와 같은 다양한 형태의 데이터를 대상으로 딥러닝을 적용하려는 시도를 하였다. 또한, 딥러닝을 적용할 수 있는 데이터의 특성을 파악하기 위해 데이터 수집형태를 달리 적용했다. 그 데이터 형태는 딥러닝을 위해 수집된 데이터, 딥러닝을 고려하지 않고 수집된 데이터, 정부가 수집해 공개한 데이터이다.

첫 번째 연구는 체압분포 데이터로부터 앉은 자세를 감지하는 것이다. 허리 건강은 앉은 자세 습관과 밀접하므로, 어렸을 때 좋은 앉은 자세를 갖게 하는 것이 중요하다. 어린이를 대상으로 통제된 환경에서 7가지 자세에 따른 압력분포 이미지 데이터가 수집되었다. 자세 분류에 사용한 딥러닝 방법은 합성곱 신경망(CNN)이며, 로지스틱 회귀 (logistic regression)와 그 분류 성능을 비교하였다. 그 결과, 전체 분류 성능에서 CNN이 로지스틱 회귀보다 20%가량 향상을 보여주었다.

두 번째 연구는 조립 공정 영상으로부터 작업 위해도 평가 결과를 도출하는 것이다. 딥러닝을 위해 준비된 데이터가 아닌, 작업 위해도 평가를 위해 촬영되었던 영상 데이터와 평가 결과를 대상으로 하였다. 작업 위해도 평가를 위해 사용되는 OWAS, RULA, REBA 세 가지 평가 방법에 딥러닝 방법인 LSTM을 적용하여 그 성능을 비교하였다. 그 결과, 딥러닝으로 OWAS 평가를 했을 때, RULA, REBA

에 비해 좋은 성능을 보여주었다.

세 번째 연구는 손의 여러 치수로부터 키를 추정하는 것이다. 정부 단위로 조사하여 공개한 데이터를 대상으로 하였다. 기존 연구에서는 선형회귀를 이용하여 손의 수치로부터 키를 추정하였다. 이에 본 연구에서는 딥러닝 방법인 RNN과 재귀적 일반화 선형 모형 (RGLM)을 적용하여 그 추정 성능을 비교하였다. 그 결과, RGLM과 RNN은 선형회귀에 비해 좋은 성능을 보여주었다.

세 연구를 통해, 딥러닝 방법이 기존의 연구 방법을 대체할 수 있음을 확인하였다. 절대적인 성능이 좋지는 않았지만, 기존 방법보다 상대적으로 좋은 성능을 보여주었다. 데이터 형식에 따라 적용할 수 있는 딥러닝 방법이 달랐으며, 딥러닝 방법에 따라서도 성능 차이가 발생했다. 다양한 케이스에 대해 학습이 되지 않은 경우, 누락된 부분에 대해서는 결과를 도출하지 못했다. 따라서, 딥러닝 적용에는 데이터 선별 및 가공이 선행되어야 한다. 인간공학 연구에 있어서, 인간공학 연구 결과물이 딥러닝을 통해 현실에 쉽게 반영될 수 있을 것이다. 딥러닝은 연구자를 대체하는 것이 아니라 연구 대상 범위와 활용 범위를 넓혀줄 것이다.

주요어: 인간공학, 딥러닝, 자세인식, 동작분석, 회귀모형
학번: 2010-21121